2004

Analysis of multi-branch torsional vibration for design optimization

Yuwen Yao
West Virginia University

Analysis of Multi-Branch Torsional Vibration For Design Optimization

Yuwen Yao

Dissertation Submitted to the
College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
in
Mechanical Engineering

Dr. Kenneth Means, Committee Chairperson
Dr. Donald W. Lyons
Dr. Victor Mucino
Dr. Wenguang Wang
Dr. Ziqing Zhuang

Department Mechanical and Aerospace Engineering

Morgantown, West Virginia
2004

Keywords: Torsional Vibration, Multi-branch System

ABSTRACT

Analysis of Multi-Branch Torsional Vibration For Design Optimization

By Yuwen Yao

Industries worldwide are rapidly developing advanced complex machinery. One area that must be considered in these engineering systems is torsional vibration for multi-branch and multi-junction systems. If torsional vibrations are not considered, they could lead to early failures and costly repairs to machinery.

Torsional vibration is a type of severe twisting motion due to improperly designed rotation machinery. It usually causes noticeable sound disturbances and potential fatigue problems. Torsional vibration happens when an excitation frequency gets close to the natural frequency of the system. This frequency exists at one or more periods of the operating range in torsional systems. Torsional vibration damage can be controlled if critical speeds or torsional natural frequencies are determined in the design stage.

This research studied an analytical model and method of predicting speed-related excitation frequencies of complex rotating systems. Also, the conditions of damping and excitation forces for multi-junction, multi-branch torsional vibration systems were discussed. A computer program was developed and verified with actual engineering examples. This model made it possible to calculate the natural frequencies and mode shapes of multi-branch torsional vibration systems with one or more junction points. A user-friendly graphic interface for modeling was presented. This study also illustrated comparisons between the analytical results and some given examples as well as results from references.

ACKNOWLEDGEMENTS

The author wishes to express his gratitude to Professor Kenneth H. Means for his support and guidance throughout the author's graduate studies; Professor Means's enormous encouragement and patience have always been the most valuable experiences the author has ever had. He is also thankful to his major professors: Donald W. Lyons, Victor H. Mucino, Wenguang Wang and Ziqing Zhuang, for their expert advice and valuable inputs throughout the dissertation.

Special thanks to Dr. Wenguang Wang and Dr. Q. K. Yuan for their generous exchange and verification of data.

The author also expresses appreciation to his wife, Nicole Yao, for her patience and devotion during the long period of time used in writing this dissertation.

Without the support of the above, this work would have never been completed.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Analysis Method and Program for Design Optimization of Multi-Branch Torsional Vibration

A multi-branch torsional vibration system is found in machinery with rotation or drive-train systems, such as mining machinery, petroleum and chemical machinery, steel rolling machinery and automobiles[1-3]. For example, such kinds of multi-branch torsional systems are found in coupled engine installations using both turbines and reciprocating engines. The rapidly growing field of screw compressors is a new addition to the similar multi-branch systems of marine installations, auto differentials, and other geared installations. For instance, recent advances in the shipbuilding industry have resulted in the construction of large and powerful vessels with extremely complex propulsion systems. These systems frequently are multi-branched, having two or more drive units.

Improper design parameters for machinery will cause serious torsional vibration. Tortional vibration occurs when an excitation frequency gets close to the natural frequency of the system. This will lead to noticeable sound disturbances, severe shaking, and component fatigue problems. For example, it is a commonly known fact that crankshaft failures can occur in internal combustion engine driven installations when the operational speed range contains significant torsional critical speeds. Because of the pulsating natural of the gas pressure in the cylinder and the inertia of the reciprocating parts, severe torsional stresses can develop in the main shafting. The result is either reduction of shaft life or fatigue failure. In order to avoid fatigue failure of crank shafts in such capital intensive machinery, it is essential that the following facts should be carried out at the design stage: calculation of natural frequencies and modes, harmonic analysis of excitation torques, selection of critical speeds and severe orders, and calculation of maximum torsional stresses in operating speed range.

Torsional vibration usually exists at one or more periods of the operating range in torsional systems. It is very important to analyze and pre-estimate critical speeds or torsional natural frequencies and mode shapes of the vibration systems in the design stage. This way, future disastrous and costly repairs of the machinery will be controlled. The study of analysis theory and methods of multi-branch torsional vibrations is becoming an important subject as the complexity of modern machines increases. The analysis of multi-branch torsional vibration grows more and more important in industries moving toward large scale systems, high speeds and automation. These include mining, rolling steel, oil and chemical, machinery and shipbuilding industries.

Besides, a mechanical power transmission system is usually one part of a machine, which is often subjected to static or periodic torsional loading that necessitates the analysis of the torsional characteristics of the system. For example, the drive train of a typical automobile is subjected to a periodically varying torque[2]. This torque variation occurs due to the cyclical nature of the internal combustion engine that supplies the power[3]. If the frequency of the engine's torque variation matches one of the resonance frequencies of the engine/drive train system, large torsional deflections and internal shear stresses can occur. Continued operation of the machinery under such conditions can lead to early fatigue failure of system components.

Thus, an engineer designing such a system needs to be able to predict its torsional natural frequencies and to easily determine what effects design changes might have on those natural frequencies.

An efficient and accurate method and program for predicting torsional natural frequencies of a piece of machinery should be capable of modeling the important characteristics of the system in a timely manner. Accurately modeling a system in the early stages of a design can reduce costs by decreasing the number of changes needed at later stages in the design process. In the case of modeling torsional system characteristics, it is common to find machinery with vibration dampers, tuned absorbers, and multiple shafts connected by gear trains that can significantly affect the system's dynamic performance. An accurate model of the system must be flexible enough to account for such components. However, a balance must be maintained between the accuracy of the model being created and the amount of time and effort needed to create the model. Therefore, a valuable design tool for torsional analysis would allow the engineer to quickly create a model of the system that provides insight about the system characteristics.

1.2 Literature Review

Many skilled researchers have conducted extensive investigations in this field. However, the current studies on multi-branch vibration systems are essentially an extension of traditional theory and methods for some particular cases. This is especially true for Holzer's method and the transfer matrix method. The transfer matrix method for determining natural frequencies of torsional systems is an extension and the matrix form of the Holzer method in which the equations relating the displacements and internal forces of the system are written in matrix form.

Wilson [4] gives a historical review of the early development of modern torsional analysis. It is reported that failures in marine and aeronautical drive trains were the original source of interest in the dynamic torsional behavior of machinery.

Nestorides [5] describes methods for modeling the various elements of torsional systems. These references include methods for determining equivalent inertias and/or stiffnesses for a variety of machinery components including crankshafts, flywheels, couplings, absorbers, etc. It is common for machinery systems to consist of multiple shafts geared together in non-branched or branched systems. Both references describe a method for modeling non-branched, multi-shaft systems as an equivalent single-shaft system as well as a procedure for performing Holzer method calculations for branched systems.

Pestel and Leckie [6] describe the transfer matrix technique for analyzing a branched-torsional system, which involves reducing the branched system to an equivalent single-shaft system. This method requires lumping the characteristics of the branch at the point on the main system where the branch is attached. That technique requires the elimination of the branch's state relations from the global transfer matrix and can result in numerical difficulties when using a root finding routine to determine natural frequencies. These numerical difficulties result from infinity wraps that can be observed by plotting the characteristic determinant curve for a branched system. The transfer matrix method can be used to a wide variety of problems including the determination of natural frequencies and mode shapes for undamped and damped torsional systems. In the process of determining the eigenvalues of a torsional system or the system's response to a torsional excitation the boundary conditions of the model must be applied.

Pilkey and Chang [7] present a generalized method for applying the boundary conditions to a torsional transfer matrix model that is useful in developing an algorithm to accomplish the desired analysis. Pilkey and Chang also present a number of useful torsional transfer matrices and describe a computer program, TWIST, capable of performing torsional analysis for branched systems.

Tavares and Prodonoff [8] presents a new modelling procedure for using in analyses of torsional vibration of gear-branched propulsion systems, which has evolved from considerations on the use of constrained finite element equilibrium equations.

Shaikh [9, 10] developed a general and direct method for the analysis of branched systems, in which transfer matrices were used in Holzer-type solutions. He considered that the method should be applicable not only for torsional vibration systems but also for other branched systems. In this method, no matrix inversions (or equivalent operations) were required to account for branches at a junction. A single determinant giving natural frequencies was reached irrespective of the number of branches and junctions. Thus, the method is straightforward, compact, and economical for computer solutions.

Dawson and Davies [11] developed a globally convergent iteration technique for application to residual function value vibration analysis methods as an extension to the method proposed by Shaikh. This method yielded a fully automatic, efficient method regardless of the natural frequency distribution or frequency range of the problems. The iteration formula in the extended method required the first and second derivatives of the residual determinant as well as the determinant itself. The method of derivation of these derivatives via both a matrix transfer and Holzer procedure was presented. Illustrative examples of the application of the extended method to the solution of the torsional natural frequencies of marine geared drive systems were presented which demonstrated the power and efficiency of the extended method, irrespective of the natural frequency distribution or the frequency range of the problem.

Eshleman [12] used the transfer matrix method to build up a refined mathematical model of the engine and end item power shafts. He utilized the model to determine their natural frequencies, mode shapes, torsional motions and stresses. The mathematical model is composed of a finite number of elements which simulate lengths of continuous, massive, elastic shaft with end attached lumped mass and springs.

Sankar [13] presents one multi-shaft torsional transfer matrix approach that maintains the state information for the entire model in the global transfer matrix. This method involves building the transfer matrix for each branch separately, applying compatibility relations at the junction where the branches join, and then using the boundary conditions to find the characteristic determinant of the system. However, that method is cumbersome for complicated systems with multiple branches.

Sankar [14] developed a new method based on the extended transfer matrix method to analyze free vibration of multi-branch torsional vibration systems. The method was radically different from the traditional methods in that an extended transfer matrix relation was formulated for each branch. For this, the calculations were propagated from the junction and proceed simultaneously in all branches toward their respective ends. Then by substituting the compatibility and equilibrium conditions, a frequency dependent characteristic matrix was formulated. This procedure reduced the size of the matrix and automatically eliminated the need of any additional operation such as matrix inversion and the solution of a system of equations for the formulation of the characteristic matrix. Finally, the boundary conditions were applied to the matrix relation and the natural frequencies were determined from the roots of a frequency determinant derived from the characteristic matrix.

Mitchell [15] has modified a multi-rotor transfer matrix approach for geared-torsional systems which was originally developed by Hibner [16] for shafts experiencing lateral vibrations. This multi-rotor transfer matrix approach is a simple and effective method for modeling multi-shaft systems. The model building procedure associated with this method can be readily generalized for application in a computer program.

Abhary [17] advocates the use of a semi-graphical approach for modeling lumped-parameter torsional systems. The graphical part of the technique is simply a bookkeeping tool to aid the analyst in performing equivalence calculations for systems with several branches. Once the equivalent model has been created, the author suggests writing the equations of motion for the system in matrix form and performing an eigenvalue analysis with the aid of a commercial software package. However, for complicated systems the necessary equivalence calculations can become time consuming and tedious. Therefore, this technique is not optimal for use in a design tool for torsional analysis.

Edwards and Gray [18-19] developed a torsional vibration analysis program suitable for handling multi-junction multi-branched systems with damping and excitation torques (provided by engines, compressors, marine propellers, etc.)

applied at many points in the individual branches. The method used a 2 by 2 matrix method for branched systems. This method was considered to be a major advancement on other methods of vibration analysis, since it is faster and cheaper for repeated use than either the current Holzer's table method, the field matrix method or the iteration methods. It was also more flexible, permitting the user to include modifications and allowed the user to ask various questions concerning the behavior of specific parts of the installation investigated, to which specific answers were provided without having to evaluate all the conditions at all points in the system. But, this method is not good for computer programming, and the problems solving process needs interference by users.

Blanding [20] describes a transfer matrix computer program that implements the Hibner/Mitchell multi-rotor transfer matrix approach for analyzing the three-dimensional, harmonically forced response of multiple-shaft systems. This three-dimensional response includes not only torsional response but also axial and lateral responses. This model includes coupling terms between the different degrees of freedom. The program has the capability to represent the time-varying stiffness of a pair of meshing spur gears. In addition gear mesh errors can also be included in the model to determine their effects on the response. These added modeling capabilities increase the program's ability to model a system accurately and as such are significant contributions to the development of the transfer matrix method for modeling rotors. However, including such advanced modeling capabilities comes at the expense of increasing the complexity of the program. Tsai and Kuang [21] also report of a computer program which implements the multi-rotor transfer technique for coupled lateral-torsional vibration analysis of geared rotors. Tsai and Kuang present an example uncoupled torsional analysis of a three-shaft system. However, some of the parameters used for modeling the system have inappropriate units. Therefore, the results they obtained cannot be used as a test case for a new computer program.

Doughty and Vafaee [22] report on a transfer matrix computer program capable of determining the damped natural frequencies and mode shapes of simple torsional systems. Two example problems with solutions are provided to demonstrate the technique. However, analysis using the program is limited to systems for which an Infinity Wraps equivalent single-shaft system can be developed. The root search method used in this case is a Newton-Raphson algorithm that has certain drawbacks. The Newton-Raphson technique requires an initial root estimate that can affect the success of the routine in finding roots. In addition this root-finding method requires the calculation of derivatives of the function being considered. For the transfer matrix problem these derivatives must be approximated in a somewhat arbitrary fashion which can also affect the success of the root-finding efforts. Huang and Horng [23] also describe a transfer matrix computer program that uses the Newton-Raphson technique for finding the roots of damped torsional systems. This program implements the Pestel and Leckie branching technique for a two-shaft system. Because this technique keeps track of only the main system state values and not those of the branch, calculating the complete eigenvectors for a two-shaft system using their program requires two separate system models.

Many other researchers have conducted extensive investigations into multi-branch torsional vibration, such as Mahalingam [24~26], Gilbert [27], Hundal [28], Dawson and Davies [29], Wang [30], Rawtani [31], Lai [32], Robert [33], Shigley and Mischke [34], Thomson and Dahleh [35], Whally and Ebrahimi [36].

Although there are some extensive investigations in this field, the current studies on multi-branch vibration systems are essentially an extension of traditional theory and methods for some particular cases. This is especially true for Holzer's method and the transfer matrix method. The above studies are also limited in their applications and most of them are not systemized. The studies must also be adapted for computer solutions and make methods adaptable to all cases of multi-branch tortional vibration.

Wang said that work is still needed to develop a more efficient and accurate method for analyzing multi-branch torsional vibration systems [37]. A few of the references gives the same suggestion, including Huang [38], Jaksic and Boltezar [39], and Mandal, Sivakumar and Kumar [40].

1.3 Goal of This Study

Main problems to be solved in current studies of multi-branch torsional vibration are as follows:

1 The current studies in this field mainly expand Holzer's method and transfer matrix method and solve only particular special engineering applications.

2 Designers are limited in program selection when solving problems. These programs are only suitable for some engineering applications, since they are not generalized.

Aiming at the above problems, this study has developed an effective and accurate analysis method for calculating the natural frequencies and mode shapes of multi-junction, multi-branch torsional vibration systems. This study has also developed a generalized program with a user-friendly graphic interface. The results of the simulations will be compared to those obtained analytically as well as those given in references.

This study addresses following problems:

1 Explored a systematic analysis theory and methods of multi-branch torsional vibration systems, as well as discussions of application conditions of existing methods.

2 Studied the influence of damping for free or forced torsional vibration systems.

3 Developed an effective and accurate analysis method for calculating natural frequencies and mode shapes of multi-junction, multi-branch

torsional vibration systems. A modified version of Holzer's method is used in this study where the calculations are propagated from branch to branch in a systematic and an efficient manner.

4 Developed a generalized user-friendly graphic interface program and use it to solve some engineering examples. Results of comparisons between the software predictions and some existing engineering applications are given in Chapter 6.

With the studied analysis theory, method, and program, designers will easily calculate natural frequencies and mode shapes of multi-branch torsional vibration systems. The simulation results can be used to guide engineers in designing products, ensuring the product quality.

Benefits to industry by the proposed theory, methods, and the developed program from this study are as follows:

1 To analyze existing machines for problems, then supervise the machines re-design.

2 To estimate the new machine's performances in a design stage to avoid future disastrous and costly repairs of the designed machines.

A complete and thorough study was accomplished by extensive research and examination. Steps included:

1 Discussed the features and application conditions of existing theory and methods.

2 Developed a new, efficient and accurate method to calculate torsional natural frequencies of complex rotating systems.

3 Studied the influence of damping for free or forced torsional vibration

systems.

4 Developed a software system based on the proposed theory and methods.

5 Verified with some typical engineering application examples of multi-branch torsional vibration problems, and comparing results.

CHAPTER 2

ANALYSIS FOUNCATION OF MULTI-BRANCH TORSIONAL VIBRATION

2.1 Basic Theory of Mechanical Vibration Analysis

A *n*-degree-of-freedom vibration system is described by a set of *n* simultaneous ordinary differential equations in second-order. The system has as many natural frequencies as the degrees of freedom. A mode of vibration is associated with each natural frequency.

Generally, solving a vibration problem takes two steps. First, formulating the motion equations of the vibration system, and then solving these motion equations and getting natural frequencies and mode shapes.

There are three main methods to formulate the motion equations of a vibration system: 1) Newton's second law method; 2) influence coefficient method; 3) energy method.

To solve the motion equations, one needs to calculate roots of frequency equations and get the natural frequencies and mode shapes.

For example, a torsional vibration system with three degrees of freedom as shown in Figure 2-1.

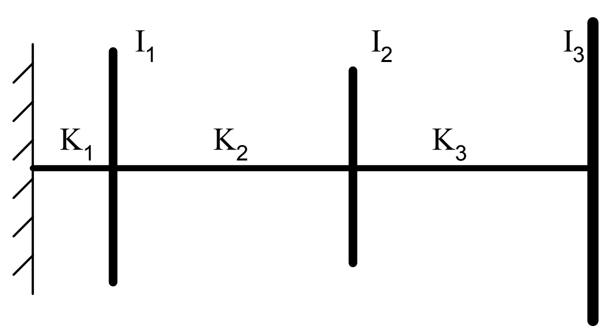

Figure 2-1. A Three Degree of Freedom Torsional System

- Step One: Formulating motion equations of the vibration system

From Newton's second law, the equations of motion are:

$$
\begin{aligned}
I_1\ddot{\theta}_1 &= -K_1\theta_1 - K_2(\theta_1-\theta_2) \\
I_2\ddot{\theta}_2 &= -K_2(\theta_2-\theta_1) - k_3(\theta_2-\theta_3) \\
I_3\ddot{\theta}_3 &= -K_3(\theta_3-\theta_2)
\end{aligned}
\tag{2-1}
$$

or

$$[I]\{\ddot{\theta}\}+[K]\{\theta\}=\{0\} \tag{2-2}$$

Substituting $\theta_i = \theta_i \sin(\omega t+\psi_i)$, for i=1,2, and 3, in equation (2-1), factoring out the $\sin(\omega t+\psi)$ term, and rearranging the above equations then, we have

$$
\begin{aligned}
&(K_1+K_2-\omega^2 I_1)\theta_1 - K_2\theta_2 = 0 \\
&-K_2\theta_1 + (K_2+K_3-\omega^2 I_2)\theta_2 - K_3\theta_3 = 0 \\
&-K_3\theta_2 + (K_3-\omega^2 I_3)\theta_3 = 0
\end{aligned}
\tag{2-3}
$$

or

$$\{[K]-\omega^2[I]\}\{\theta\}=0 \tag{2-4}$$

Where

$$[K]=\begin{bmatrix} K_1+K_2 & -K_2 & \\ -K_2 & K_2+K_3 & -K_3 \\ & -K_3 & K_3 \end{bmatrix} \tag{2-5}$$

$$[I]=\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} \tag{2-6}$$

$$[\theta]=\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\end{bmatrix} \tag{2-7}$$

- Step Two: solving these motion equations and getting the natural frequencies and mode shapes

The frequency equation can be obtained by making the determinant $\Delta(\omega)$ (the coefficients of θ_1, θ_2, θ_3) to zero.

$$\Delta(\omega)=|[K]-\omega^2[I]|=0 \tag{2-8}$$

or

$$\Delta(\omega)=\begin{vmatrix}K_1+K_2-\omega^2 I_1 & -K_2 & \\ -K_2 & K_2+K_3-\omega^2 I_2 & -K_3 \\ & -K_3 & K_3-\omega^2 I_3\end{vmatrix}=0 \tag{2-9}$$

Equation (2-9) is usually called an eigenvalue equation or frequency equation. By solving eqn. (2-9), we can get the natural frequencies.

For the reason of simplification, let $I_1=I_2=I_3=I,\quad \text{and}\quad K_1=K_2=K_3=K$, the frequency equation is:

$$\omega^6-5\left(\frac{K}{I}\right)\omega^4+6\left(\frac{K}{I}\right)^2\omega^2-\left(\frac{K}{I}\right)^3=0 \tag{2-10}$$

The roots of the equation are:

$$\{\omega^2\}=\begin{Bmatrix}0.198\dfrac{K}{I}\\ 1.55\dfrac{K}{I}\\ 3.25\dfrac{K}{I}\end{Bmatrix} \tag{2-11}$$

Substituting them into equation (2-3) or (2-4), we can obtain the mode shapes.

2.2 Basic Methods of Mechanical Vibration Analysis

In order to get the natural frequencies and mode shapes, we must solve frequency equations.

When the number of degree of freedom increases in a vibration system, it becomes more difficult to get solutions. In this case, we can use approximate numerical methods.

Approximate methods having been widely used are as follows:

1) Methods based on matrix theory, including matrix iteration method, Jacobi method, QR method, sub-space matrix iterative method and Dunkerley method.

2) Methods based on energy theory, including Rayleigh method and Ritz method.

3) Holzer's method and transfer matrix method, which are very useful means, especially for chain-type (or Hozer-type) vibration problems.

The following is a brief review of these above methods.

- Methods based on matrix theory

Methods based on matrix theory are generally used to solve most of the vibration problems. These methods need matrix transformation or matrix iteration, and its calculation is complex.

Dunkerley’s method is used only for estimating fundamental frequency of the vibration system. The estimated fundamental frequency is always lower than the exact value, since the harmonics are neglected in the equation.

Dunkerley's method comes from the method of influence coefficients. Consider the free vibration of an undamped system, by the method of influence coefficients, we have

$$\{q\} = [d_{ij}]\{-m\ddot{q}\} \tag{2-12}$$

where $\{q\}$ is the displacement vector, $[d_{ij}]$ the flexibility matrix, and $\{-m\ddot{q}\}$ the vector of inertia factors. At a principal mode of vibration, the deflections $\{q\}$ are harmonic with $\{\ddot{q}\} = \{-\omega^2 q\}$. Substituting this in the equation above gives

$$\{q\} = [d_{ij}]\{\omega^2 mq\} \tag{2-13}$$

Dunkerley's equation is deduced from Equation (2-13) by retaining only the fundamental frequency.

From Equation (2-13), we can get

$$\frac{1}{\omega_1^2} + \frac{1}{\omega_2^2} + \cdots + \frac{1}{\omega_n^2} = \frac{1}{\omega_{11}^2} + \frac{1}{\omega_{22}^2} + \cdots + \frac{1}{\omega_{nn}^2} = \sum_{i=1}^{n} \frac{1}{\omega_{ii}^2} \tag{2-14}$$

where ω_{ii} is the natural frequency of an equivalent mass spring system with m_i acting along at station i.

If the fundamental frequency ω_1 is much lower than that of the harmonic ω_2, we have

$$\frac{1}{\omega_1^2} >> \frac{1}{\omega_2^2} \tag{2-15}$$

and

$$\frac{1}{\omega_1^2} \approx \sum_{i=1}^{n} \frac{1}{\omega_{ii}^2} \tag{2-16}$$

which is Dunkerley's equation.

Matrix iteration method yields one frequency and one mode vector at a time. It first iterates for the lowest (or the highest) frequency and mode. Then this mode is used as a base and the same process is repeated to obtain the next mode. By using this method, it can obtain a few frequencies and mode shapes from the fundamental frequency. A server limitation of matrix iterative method is an accumulation of errors as we proceed towards to the estimation of the frequency of the second and higher modes.

- Methods based on energy theory

Rayleigh method is commonly used for estimating the fundamental frequency of the vibration system. If an exact mode shape is assumed, the frequency calculated will be exact. If the assumed mode shape is not an exact dynamic mode shape, it is equivalent to the application of additional constraints to the vibratory system. Hence, the calculated frequency is higher than the true value. Thus, the Rayleigh method tends to give a higher value for estimated frequency. The estimated error depends on the distance of the assumed first mode shape from the exact mode shape.

Ritz method is similar to Rayleigh method, and the assumption for the fundamental mode shape is closer to the exact value.

- Methods especially suitable to chain-type (or Holzer-type) vibration problems

Holzer's method is essentially a systematic tabulation of frequency equation of a system. The method has general applications, including systems with rectilinear and angular motions, damped or undamped, semi-defined, branched, or systems with fixed ends. The procedure can be programmed for computer applications.

The method assumes a trial frequency at the beginning. A solution can be derived when the assumed frequency satisfies the constraints of the problem. Usually, this requires several trials. Depending upon the trial frequency used, the fundamental as well as the harmonic frequencies can be determined. The tabulation also gives the mode shape of the system.

In the transfer matrix method, the shaft elements are considered as massless springs with torsional stiffness and the disks are considered as lumped masses. A transfer matrix is developed for each segment to relate the state variables across the element or station. The transfer matrix method is a step-by-step procedure where the state vector at one point is related to the state vector at another point by a matrix. The boundary conditions determine the value of the state vector at each end of the chain. Starting at one end of the chain by a successive process of matrix multiplication, we establish the state vector at the other end of the chain. Having thus established state vector, we can determine the frequency and the mode shape of the oscillation. Each degree of freedom requires only an additional step of matrix multiplying. This method is ideal for linear one dimensional models, such as rotors of an electric motor on multi-bearing supports.

In summary, all above methods may be used for analyzing multi-branch torsional vibration. But, methods based on matrix theory and methods based on energy theory are general methods. For multi-branch torsional vibration problems, these methods are usually limited by application conditions. Therefore, they are not yet commonly employed in analyzing multi-branch torsional vibration systems.

Holzer's method and the transfer matrix method are the two basic methods for this type of torsional vibration (in-line systems). These methods are the most efficient and accurate to calculate torsional natural frequencies and mode shapes and can be programmed by computer languages.

The theory of Holzer's method is the same as the transfer matrix method. In the Holzer method, the shaft elements are considered as massless springs with torsional stiffness and the disks are considered as lumped masses. A transfer matrix is developed for each segment to relate the state variables across the element or station. Thus, Holzer's method utilizes transfer matrices to propagate the effect of assumed and known boundary conditions through a *N* station system model. Frequency is an independent variable from which the mode shape can be determined. If the mode shape obtained satisfies the boundary conditions, then the assumed frequency is a natural frequency. Both trial-and error and iteration procedures have been used in determining the frequency.

Therefore, this study will use Holzer's method and the transfer matrix as basic methods and also make a new contribution to these methods to solve multi-branch torsional vibration problems.

The principles of these two methods are discussed in the following section.

2.3 Basic Principle of Holzer's Method

Consider a three-disk torsional vibration system shown below (Figure 2-2).

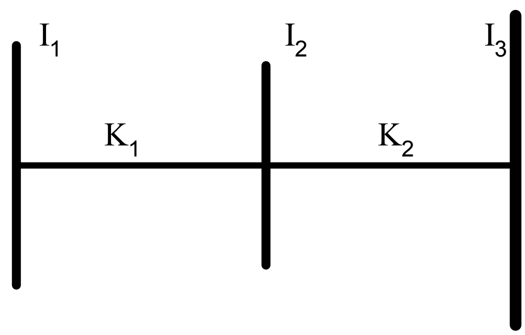

Figure 2-2 A Three-Disk Torsional Vibration System

The motion equations from Newton's second law are:

$$
\begin{aligned}
I_1\ddot{\theta}_1 &= -K_1(\theta_1 - \theta_2) \\
I_2\ddot{\theta}_2 &= -K_1(\theta_2 - \theta_1) - K_2(\theta_2 - \theta_3) \\
I_3\ddot{\theta}_3 &= -K_2(\theta_3 - \theta_2)
\end{aligned}
\tag{2-17}
$$

The motions are harmonic at a principal mode of vibration. Substituting θ_i, $i = 1, 2, 3$, into equation (2-17) by:

$$\theta_i = \theta_i \mathrm{Sin}(\omega t + \psi) \tag{2-18}$$

Simplifying the above equations

$$\begin{aligned}
&-\omega^2 I_1\theta_1 + K_1(\theta_1 - \theta_2) = 0 \\
&-\omega^2 I_2\theta_2 + K_1(\theta_2 - \theta_1) + K_2(\theta_2 - \theta_3) = 0 \\
&-\omega^2 I_3\theta_3 + K_2(\theta_3 - \theta_2) = 0
\end{aligned} \tag{2-19}$$

Make equation (2-19) in index form

$$\sum_{i=1}^{3} I_i\theta_i\omega^2 = 0 \tag{2-20}$$

Correspondingly, for a *n* disk system

$$\sum_{i=1}^{n} I_i\theta_i\omega^2 = 0 \tag{2-21}$$

Equation (2-21) states that the summation of the inertia torque of a vibration system must be zero. The trial frequency ω must satisfy this constraint. Hence, equation (2-21) is another form of the frequency equation.

To begin the tabulation, first, assume a trial frequency ω and let θ_1=1, arbitrarily, then calculate θ_2 from the first equation in equation (2-19). Then θ_3 from the second equation

$$\begin{aligned}
&\theta_1 = 1 \\
&\theta_2 = \theta_1 - \frac{\omega^2 I_1\theta_1}{K_1} \\
&\theta_3 = \theta_2 - \frac{\omega^2 (I_1\theta_1 + I_2\theta_2)}{K_2}
\end{aligned} \tag{2-22}$$

Substitute the values of θ_1 , θ_2 and θ_3 into equation (2-19) or (2-20) or (2-21) to check if the constraint is satisfied. If not, make a new assumption of ω, and repeat the process until the constraint is met.

According to equation (2-22), we can reduce the general expressions of angular deformations for a *n* disk system as

$$\theta_j = \theta_{j-1} - \frac{\omega^2 \sum_{i=1}^{j-1} I_i \theta_i}{K_{j-1}} \qquad j = 1,2,\ldots,n \tag{2-23}$$

In summary, this method consists of the repeated application of equations (2-21) and (2-23) for different trial frequencies. If the trial frequency is not a natural frequency of the system, equation (2-21) will not be satisfied. In this case, the left part of equation (2-21) is called residual torque, which can be expressed as follows:

$$T_n = \sum_{i=1}^{n} I_i \theta_i \omega^2 \tag{2-24}$$

The residual torque *T*, represents a torque applied at the last disk. It is equivalent to a condition of steady-state forced vibration. A typical residual torque *T* versus ω plot can be expressed in Figure 2-3.

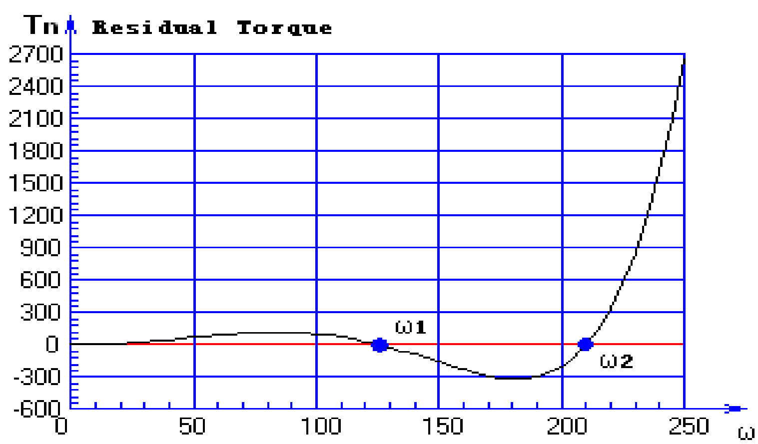

Figure 2-3 $T-\omega$ Curve (Residual Torque Versus ω)

Considering another example, a three-disk system with one fixed end, shown in Figure 2-4. The constraint is that the displacement at the fixed end must be zero. With slight modifications, the procedure above can be applied to this case.

In this case, the three-disk system shown in Figure 2-4 can be considered as a four disks system. The last disk is disk 4 and its mass is infinite.

The calculating formations are as following:

Let $$\theta_1 = 1.0 \tag{2-25}$$

$$T_k = \sum_{i=1}^{K} I_i \theta_i \omega^2 \qquad (k = 1,2,3) \tag{2-26}$$

$$\theta_{k+1} = \theta_k - \frac{T_k}{K_k} \qquad (k = 1,2,3) \tag{2-27}$$

Then find out the frequencies ω_i (refer 2-23) which satisfied the condition of $\theta_4 = \theta_3 - \frac{T_3}{K_3} = 0$. These are the natural frequencies. See Figure 2-5.

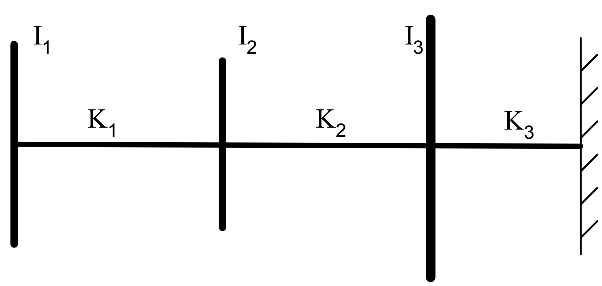

Figure 2-4 A Torsional System with One Fixed End

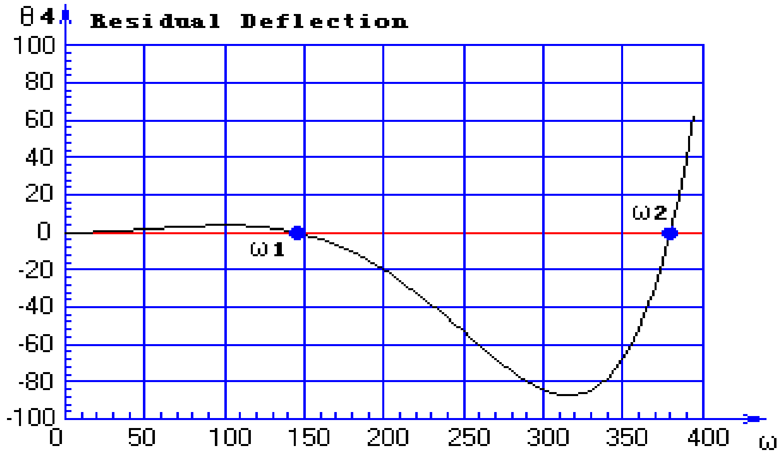

Figure 2-5 θ - ω Curve (Residual Deflection Versus ω)

CHAPTER 3

ANALYTICAL THEORY AND METHODS OF MULTI-BRANCH TORSIONAL VIBRATION

This study has explored an analysis theory and methods of multi-branch torsional vibration systems, and discussed the features, efficiencies and applicable conditions of the existed theory and methods.

The goal of this study is to develop an effective and accurate, analysis method to calculate the natural frequencies and mode shapes of multi-branch torsional vibration systems.

Based on this method, we shall develop a generalized analysis program based on the above theory and method. This easy to use program will employ a user-friendly graphic interface for modeling the multi-branch torsional vibration systems with one or more junctions. It will automatically be able to identify the structure of a vibration system and build motion equations of the system. Users will merely input data.

The theoretical works to achieve the goal can be described as the following four parts:

1) Equivalent Shaft System

A method to change an actual vibration system into an equivalent system is presented. A calculating formula is derived.

2) One-Shaft System With Multi-Rotors

This part is the theory base of this development. The calculating method, modeling method of motion equations based on Lagrange Equation and solving method of motion equations are described. Finally, the formula and procedure of Holzer's method are derived, which supply a basic idea for the following sections.

3) Multi-Branch Torsional Vibration System

The studies in the above section are expanded into multi-branch (or multi-shaft) systems. The mathematical formula and calculating procedure for these kinds of systems are derived.

4) Multi-Junction, Multi-Branch Torsional Vibration System

In this part, the studies in the above sections are further expanded for multi-junction, multi-branch systems. The mathematical formula and calculating procedure for the systems with multi-junction and multi-branch are derived.

3.1 Equivalent Shaft System

To analyze an actual multi-branch system, usually, the first thing is to change the system into an equivalent system. Considering the two-shaft geared system shown in Figure 3-1:

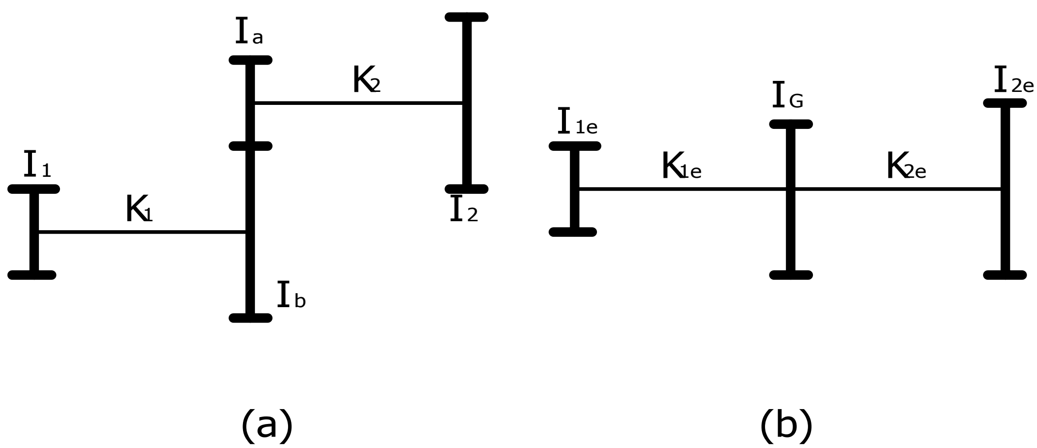

Figure 3-1 Two Shaft with Gears and Its Equivalent System

The effective stiffness and inertia for one shaft relative to the other can be calculated for shafts operating on two different speeds. To analyze a gear system, one of the shafts is chosen as a reference for the rotational speed, and the other gear shaft is referred to the first chosen shaft speed.

In Figure 3-1 part (a), I_1 and I_2 are the gears, I_a, I_b are the other rotors. Part (b) shows an equivalent shaft system. For an equivalent system to truly represent the actual system, the kinetic energy and the strain energy of the two systems must always be equal. Thus, if the speed ratio N, and shaft 1 rotates at ω_1 and shaft 2 rotates at ω_2, then:

$$\frac{\omega_2}{\omega_1} = \frac{\theta_2}{\theta_1} = N$$

$$\omega_2 = N\omega_1 \qquad \theta_2 = N\theta_1 \tag{3-1a}$$

If shaft 1 is chosen as reference, then, in the equivalent system,

$$\omega_e = \omega_1,\ \ \theta_{1e} = \theta_{2e} = \theta_1,\ \ I_{1e} = I_1,\ \ K_{1e} = K_1 \tag{3-1b}$$

Where $\theta_{1e},\ \theta_{2e},\ \theta_1$ are the motion equations of the shafts.

According to the equivalent theory, the kinetic energy and strain energy of an equivalent system should be equal to the kinetic energy and strain energy of the original system.

The kinetic energy for the original engineering system shown in Figure 3-1(a),

$$\frac{1}{2}I_1\omega_1^2 + \frac{1}{2}I_b\omega_1^2 + \frac{1}{2}I_a\omega_2^2 + \frac{1}{2}I_2\omega_2^2 \tag{3-2a}$$

Substitute Equation (3-1a) to Equation (3-2a), we can obtain,

$$\frac{1}{2}I_1\omega_1^2 + \frac{1}{2}(I_b + I_aN^2)\omega_1^2 + \frac{1}{2}I_2N^2\omega_1^2 \tag{3-2b}$$

The kinetic energy for the equivalent system shown in Figure 3-1(b),

$$\frac{1}{2}I_{1e}\omega_e^2+\frac{1}{2}I_G\omega_e^2+\frac{1}{2}I_{2e}\omega_e^2 \tag{3-2c}$$

Substitute Equation (3-1b) to Equation (3-2c), we can obtain, we can obtain,

$$\frac{1}{2}I_{1e}\omega_1^2+\frac{1}{2}I_G\omega_1^2+\frac{1}{2}I_{2e}\omega_1^2 \tag{3-2d}$$

Comparing Equation (3-2d) and (3-2b), we will get

$$I_{2e}=N^2I_2\,,\quad I_G=I_b+N^2I_a \tag{3-2e}$$

The strain energy for the original engineering system shown in Figure 3-1(a),

$$\frac{1}{2}K_1\theta_1^2+\frac{1}{2}K_2\theta_2^2 \tag{3-3a}$$

Substitute Equation (3-1a) to Equation (3-3a), we can obtain,

$$\frac{1}{2}K_1\theta_1^2+\frac{1}{2}K_2(N\theta_1)^2=\frac{1}{2}(K_1+N^2K_2)\theta_1^2 \tag{3-3b}$$

The strain energy for the equivalent system shown in Figure 3-1(b),

$$\frac{1}{2}K_{1e}\theta_{1e}^2+\frac{1}{2}K_{2e}\theta_{2e}^2 \tag{3-3c}$$

Substitute Equation (3-1b) to Equation (3-3c), we can obtain,

$$\frac{1}{2}K_{1e}\theta_{1e}^2+\frac{1}{2}K_{2e}\theta_{2e}^2=\frac{1}{2}(K_{1e}+K_{2e})\theta_1^2 \tag{3-3d}$$

Comparing Equation (3-3d) and (3-3b), we will get

$$K_{1e}=K_1\,,\quad K_{2e}=N^2K_2 \tag{3-3e}$$

In equations (3-2e) and (3-3e), all stiffness and moment of inertia of the original system are replaced by dynamically equivalent ones, all of which will refer to the speed of shaft 1. The original 4-mass system at two shafts is then reduced to a 3-mass system at one shaft.

3.2 One Shaft System With Multi-Rotors

Analysis theory and method of a one-shaft system with multi-rotors, will be employed as a base for multi-junction, multi-branch torsional vibration systems.

Considering a single shaft torsional vibration system shown in Figure 3-2.

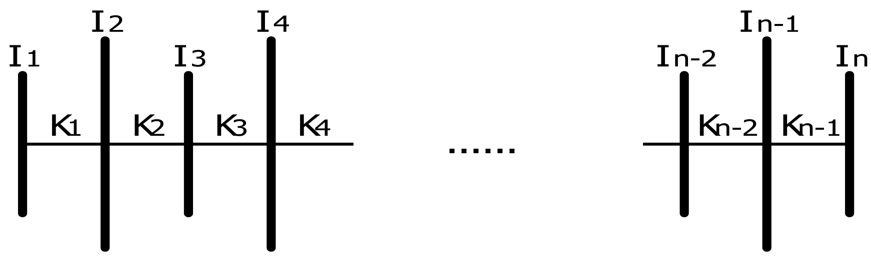

Figure 3-2 Torsional Vibration In-line Systems with Multi-Rotors

To solve the above torsional vibration system, a common approach is to approximate the actual system as an equivalent discrete system. The moments of inertia of the attached masses and the shaft elements between two nodal points are summed up and lumped at a chosen location corresponding to the approximate center of gravity.

The attached masses, m, and its moments of inertia can be calculated as follows:

$$m = \frac{\gamma}{g}\frac{\pi}{4}(D^2 - d^2)L \tag{3-4}$$

$$I_G = \frac{\gamma}{g}\frac{\pi}{32}(D^4 - d^4)L \tag{3-5}$$

where g is acceleration due to gravity; γ is specific gravity; *D, d* is outer and inner diameter, respectively*; L* is the length.

From equations (3-4) and (3-5), if the mass is known, moments of inertia also can be calculated according to the following formula:

$$I_G = \frac{m}{8}(D^2 + d^2) \tag{3-6}$$

Assuming that the concentrated masses are connected by weightless springs, then the approximate stiffness, K_j, of shafts between these masses can be calculated as follows:

$$K_j = \frac{JG}{L} \tag{3-7}$$

where *J* is polar area moment of inertia; *G* is the shear modulus of elasticity.

$$J = \frac{\pi d^4}{32}, \quad \text{and} \quad \mathrm{G} = \frac{\mathrm{E}}{2(1+\nu)}$$

where *E* is the modulus of elasticity; ν is Poisson's ratio.

In deriving the equation of motion, the kinetic energy T and the potential energy V can be obtained from the following two equations:

$$T = \frac{1}{2} I_1 \dot{\theta}_1^2 + \frac{1}{2} I_2 \dot{\theta}_2^2 + \ldots\ldots \frac{1}{2} I_n \dot{\theta}_n^2 \tag{3-8}$$

and

$$V = \frac{1}{2} K_1 (\theta_1 - \theta_2)^2 + \frac{1}{2} K_2 (\theta_2 - \theta_3)^2 + \cdots \tag{3-9}$$

Putting kinetic energy T and potential energy V into the Lagrange equation, the following equation is derived:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\theta}_i}\right) - \frac{\partial T}{\partial \theta_i} + \frac{\partial V}{\partial \theta_i} = Q_i \tag{3-10}$$

The equation of motion can be obtained in the following matrix notation:

$$[I]\{\ddot{\theta}\} + [C]\{\dot{\theta}\} + [K]\{\theta\} = \{F(t)\} \tag{3-11}$$

where:

[I]= matrix of mass moments of inertia

[C]= matrix of structural damping

[k]= matrix of structural stiffness

{F(t)}= excitation vector

$$[I] = \begin{bmatrix} I_1 & & & \\ & I_2 & & \\ & & \ddots & \\ & & & I_n \end{bmatrix} \tag{3-12}$$

$$[K] = \begin{bmatrix} K_1 & -K_1 & & & \\ -K_1 & K_1 + K_2 & -K_2 & & \\ & -K_2 & K_2 + K_3 & & \\ & & & \ddots & \\ & & & & K_{n-1} \end{bmatrix} \tag{3-13}$$

Since matrix [C] is assumed to be proportional to [I] and [K], then

$$[C] = \alpha[I] + \beta[K]$$

Equation (3-11) may be nonlinear in coefficient matrices [K], [C], and [I], and can vary during the analysis. Nonlinear behavior arises from such factors as material inelasticity, large deflections, element bi-linearity, etc.

For solving nonlinear equations, either the Newmark integration method in conjunction with the direct iteration technique or the Newon-Raphson procedure can be used.

For simplicity, it is assumed that the system is undamped and non excited. Thus, the homogeneous equation of motion will be:

$$[I]\{\ddot{\theta}\} + [K]\{\theta\} = 0 \tag{3-14}$$

The angular displacement can be expressed as:

$$\theta(t) = \theta_i e^{\omega^2 t} \tag{3-15}$$

By substituting equation (3-15) into equation (3-14), the problem is reduced to an eigenvalue frequencies and modes of free vibration can be represented by the following format:

$$\{[K] - \omega_i^2 [I]\}\{\theta\} = 0 \tag{3-16}$$

The modal matrix {θ} obtained for a given problem is a normalized matrix. The natural frequencies ω_i and the corresponding modal matrix {θ} with the jth mode shape in the jth column can be obtained by several methods. Holzer's method, based on the amplitude equation for a vibration system, is one of the most widely used techniques for solving this type of twisting frequencies [14,15].

For deriving iterative formula, we can spread equation (3-16) and get the following equations:

$$(K_1 - I_1\omega^2)\theta_1 - K_1\theta_2 = 0 \tag{3-17a}$$

$$-K_1\theta_1 + (K_1 + K_2 - I_2\omega^2)\theta_2 - K_2\theta_3 = 0 \tag{3-17b}$$

$$-K_2\theta_2 + (K_2 + K_3 - I_3\omega^2)\theta_3 - K_3\theta_4 = 0 \tag{3-17c}$$

……

$$-K_{i-1}\theta_{i-1} + (K_{i-1} + K_i - I_i\omega^2)\theta_i - K_i\theta_{i+1} = 0 \tag{3-18}$$

……

$$-K_{n-1}\theta_{n-1} + (K_{n-1} + K_n - I_n\omega^2)\theta_n = 0 \tag{3-19}$$

The Equation (3-18) can be written as more general style:

$$-I_i\omega^2\theta_i + K_i(\theta_i - \theta_{i+1}) + K_{i-1}(\theta_i - \theta_{i-1}) = 0 \tag{3-20}$$

The torques can be determined by analysis of the system as follows:

$$T_1 = I_1\omega^2\theta_1 \tag{3-21}$$

$$T_2 = I_1\omega^2\theta_1 + I_2\omega^2\theta_2 = T_1 + I_2\omega^2\theta_2 \tag{3-22}$$

……

$$T_n = \sum_{i=1}^{n} I_i\omega^2\theta_i \tag{3-23}$$

By substituting equations (3-21) ~ (3-23) into (3-17) ~ (3-20), the later can be arranged as:

$$\theta_2 = \theta_1 - \frac{T_1}{K_1} \tag{3-24}$$

$$\theta_3 = \theta_2 - \frac{T_2}{K_2} \tag{3-25}$$

$$\theta_4 = \theta_3 - \frac{T_3}{K_3} \tag{3-26}$$

……

$$\theta_n = \theta_{n-1} - \frac{T_{n-1}}{K_{n-1}} \tag{3-27}$$

The natural frequency, ω, and mode shape, $\{\theta\}$, can be obtained by using equations (3-21) ~ (3-27) as follows:

1 Assume a valve for ω and let $\theta_1 = 1$;

2 The value θ_2 can be determined from the torque equation (3-21) and the amplitude equation (3-24);

3 Substitute in the second amplitude equation (3-25), get the value of θ_3;

4 Repeat above steps, the value of θ_{n-1} can be obtained;

5 The value of T_n and θ_n can then be obtained by substitution in equation (3-23) and (3-27), respectively.

There are two possible boundary conditions, either fixed or free, at each end of the shaft. If the frequency is a correct selected value, the angle deflection θ_n will be zero for a fixed end while internal torque T_n will be zero for a free end.

If the last equation does not satisfy either of the above two boundary conditions, then it is necessary to return to step 1. A new value should be assigned to ω.

By successively selecting frequency values, the remained value θ_n or T_n will decrease. The corresponding frequencies are then the acceptable values of natural frequency for a principal mode.

While searching for the natural frequencies ω, we need to test the residual value of θ_n or T_n and pay attention to its variety. This test is revised after finding the first frequency, and then admonished on changes of the sign from minus to plus. This method can be used to determine all natural frequencies of the system. The corresponding modes are defined by the amplitudes for the correct set of calculations.

3.3 Multi-Branch Torsional Vibration Systems

Systems with several branches, as shown in Figure 3-3, cannot generally be reduced to an equivalent arrangement with only a single shaft. However, a dynamically equivalent system with the same number of branches can be obtained by selecting a basic branch and referring all values (moment of inertia, stiffness, etc.) of the other branches. Although any branch of the system can be chosen as a basic shaft, but the selection of a particular one may have some practical advantages.

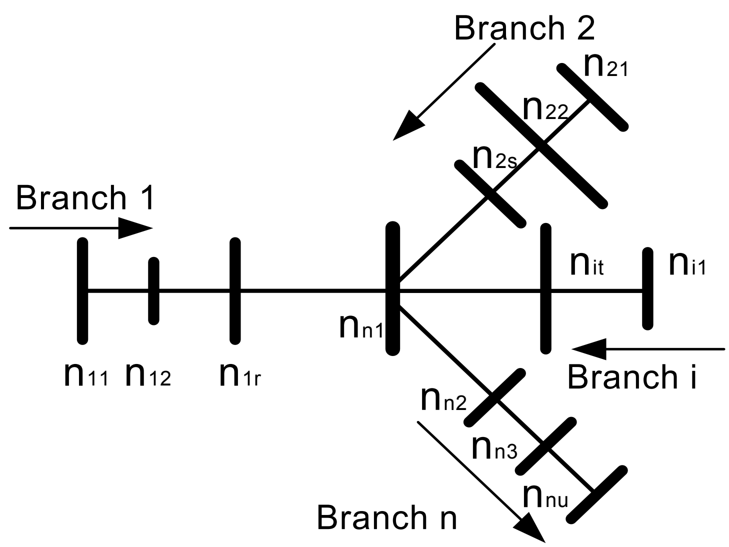

Figure 3-3 A Multi-Branch Torsional System

There are n branches in Figure 3-3, branch 1 being chosen as the basic branch. Using equations (3-21) ~ (3-27), the torsional deflection at each branch with multi-stations can be cumulatively calculated. For branches 1 through (n-1), searching direction (or direction of calculation) extends from outside end points to the junction point. For the last branch (branch n), a calculation extends from the junction point to the outside end point. In a multi-branch system, the geometrical constraints or the compatibility conditions must be satisfied.

The compatibility conditions can be described as follows:

1 At the junction point, the deflection at each branch is identical.

$$\theta_{1e} = \theta_{2e} = \theta_{3e} = \cdots = \theta_{(n-1)e} \tag{3-28}$$

where θ_{ie} expresses the deflection at the end point of branch i

2 The torque summation of the first (n-1) branches at the junction point should be equal to the starting torque at the last branch (branch n).

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} = T_{n1} \tag{3-29}$$

where T_{ie} is the torque at the end point of branch i

3 For the last branch the starting point of calculation is the junction, so that:

$$\theta_{n1} = \theta_{(n-1)e} \tag{3-30}$$

4 If the outside end is free, the torque will be equal to zero:

$$T_{11} = T_{21} = T_{31} = \cdots T_{(n-1)1} = T_{ne} = 0 \tag{3-31}$$

To satisfy the compatibility condition expressed in equation (3-28), the values of amplitude at the starting point (outside point) from branch to branch normally are not the same. That is:

$$\theta_{11} \neq \theta_{21} \neq \theta_{31} \neq \cdots \neq \theta_{(n-1)1} \tag{3-32}$$

In order to find out the proper values of these deflections θ_{i1}, a trial step is needed in the calculation. Normally assuming:

$$\theta_{11} = \theta_{21} = \theta_{31} = \cdots = \theta_{(n-1)1} = C \tag{3-33}$$

Where C is an arbitrary constant, Holzer's tabulation method is employed to obtain θ_{ie}.

Usually, $\theta_{1e} \neq \theta_{2e} \neq \theta_{3e} \neq \cdots \neq \theta_{(n-1)e}$ and the initial value of deflection at the outside end points needs to be re-determined. A new value can be calculated by the following formula:

$$\theta_{11} = \frac{\theta_{11}\theta_{1e}}{\theta_{1e}}, \quad \theta_{21} = \frac{\theta_{11}\theta_{1e}}{\theta_{2e}}, \quad \cdots, \theta_{(n-1)1} = \frac{\theta_{11}\theta_{1e}}{\theta_{(n-1)e}} \tag{3-34}$$

Using the new values of θ_{11} , θ_{21} , θ_{31} , …, $\theta_{(n-1)1}$ the deflection at the junction point will be:

$$\theta_{1e} = \theta_{2e} = \theta_{3e} = \cdots = \theta_{(n-1)e} \tag{3-35}$$

The implementation procedures for a multi-branch system are as follows:

1 Determine numbers of branch values of moment of inertia and stiffness in each branch. Set one branch as reference and input the speed ratio for each branch with regard to the reference branch.

2 Calculate the equivalent values of moment of inertia (I_i) and stiffness (K_i) to the reference branch by formula (3-2) and (3-3).

3 Calculate the values in Holzer's Table [13] for the first (n-1) branches, and the results at the junction point should meet the compatibility condition. If it is not achieved, equations (3-32) to (3-35) can be employed to recalculate the modified value. Using the modified starting value for each branch, it will be compatible at the junction point.

4 Calculate the inertial torque at the junction and continue the analysis from the junction to the outside end for the last branch. Note that equations (3-28), (3-29) and (3-30) or (3-31) must be satisfied.

5 Obtain the end point residual torque for the last branch. If the residual torque is not zero, choose another natural frequency value ω and repeat steps 1 to 5.

Plot the residual torques with the corresponding natural frequencies to get T-ω curve. This curve will be used to find the approximate frequencies to make the residual torque close to zero.

6 Use the approximate values of the natural frequencies to calculate the exact frequencies.

In this procedure, the modified linear interpolation method can be used to help find the root.

3.4 Multi-Junction Multi-Branch Torsional Vibration Systems

The above study is for calculating the natural frequencies and mode shapes of multi-branch torsional vibration systems, It is not suitable for multi-junction systems yet.

Now, we shall further extend it to solve the multi-junction, multi-branch torsional vibration problems.

Considering the multi-branch torsional vibration system with two junctions, shown in Figure 3-4.

The compatibility conditions of a multi-junction, multi-branch torsional vibration system can be described as follows:

1 At the first junction point, an angular deflection at each branch (from Branch 1 to Branch n-1) is identical.

$$\theta_{1e} = \theta_{2e} = \theta_{3e} = \cdots = \theta_{(n-1)e} \tag{3-36}$$

Where θ_{ie} expresses the deflection at the end point of branch i (i=1,2,..., n-1)

For the Link Branch (Branch n), the starting point of calculation is the first junction, so that:

$$\theta_{n1} = \theta_{(n-1)e} \tag{3-37}$$

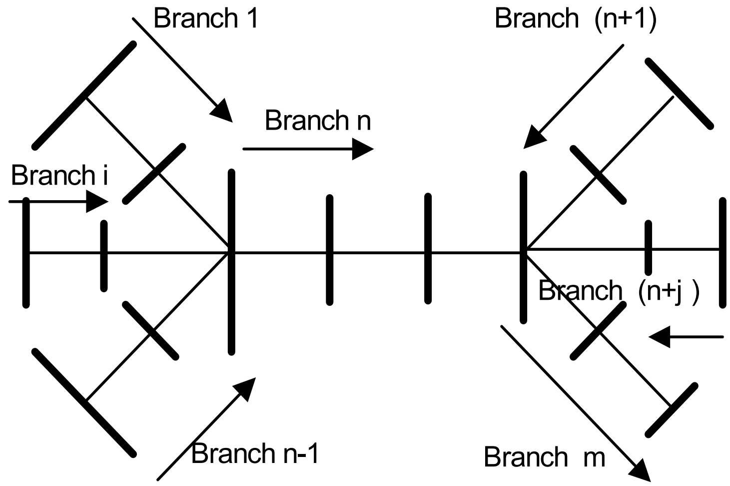

Figure 3-4 A Multi-Junction Multi-Branch Torsional Vibration System

2 At the second junction point, an angular deflection at each branch (from Branch n to Branch m-1) is identical.

$$\theta_{ne} = \theta_{(n+1)e} = \theta_{(n+2)e} = \cdots = \theta_{(m-1)e} \quad (3\text{-}38)$$

Where θ_{je} expresses the deflection at the end point of branch j (j=n, n+1, n+2, ..., m-1)

For the last branch (Branch m), the starting point of calculation is the second junction, so that:

$$\theta_{m1} = \theta_{(m-1)e} \quad (3\text{-}39)$$

3 The torque summation of the first (n-1) branches at the first junction point should be equal to the starting torque at the link branch.

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} = T_{n1} \quad (3\text{-}40)$$

Where T_{ie} is the torque at the end point of branch i (i=1,2,3,...,n-1)

4 The torque summation of branches from n to (m-1) at the second junction point should be equal to the starting torque at the last branch (Branch m), i.e:

$$T_{ne} + T_{(n+1)e} + T_{(n+2)e} + \cdots + T_{(m-1)e} = T_{m1} \qquad (3\text{-}41)$$

Where T_{je} is the torque at the end point of branch j (j=n, n+1, n+2, n+3, ..., m-1)

5 If the outside end is free of rotation, the torque will be equal to zero:

$$T_{11} = T_{21} = T_{31} = \cdots = T_{(n-1)1} = 0 \qquad (3\text{-}42)$$

$$T_{(n+1)1} = T_{(n+2)1} = T_{(n+3)1} = \cdots = T_{(m-1)1} = 0 \qquad (3\text{-}43)$$

The procedures to calculate θ_{me} and T_{me} for a multi-junction, multi-branch system are as follows:

1 Use the method described in section 2.3 to calculate θ_{n1}, T_{n1}, θ_{ne} and T_{ne};

2 Let $\theta_{(n+1)1} = \theta_{(n+2)1} = \cdots = \theta_{(m-1)1} = \theta_{n1}$

3 Calculate $\theta_{(n+1)e}$, $\theta_{(n+2)e}$, $\cdots$, $\theta_{(m-1)e}$ and

$T_{(n+1)e}$, $T_{(n+2)e}$, $\cdots$, $T_{(m-1)e}$

4 If the $\theta_{(n+1)e}$, $\theta_{(n+2)e}$, $\cdots$, $\theta_{(m-1)e}$ cannot satisfy equation (3-38), then let

$$\theta_{(n+1)1} = \frac{\theta_{n1}\theta_{ne}}{\theta_{(n+1)e}} \qquad (3\text{-}44)$$

$$\theta_{(n+2)1} = \frac{\theta_{n1}\theta_{ne}}{\theta_{(n+2)e}} \qquad (3\text{-}45)$$

$$\theta_{(n+3)1} = \frac{\theta_{n1}\theta_{ne}}{\theta_{(n+3)e}} \qquad (3\text{-}46)$$

……

$$\theta_{(m-1)1} = \frac{\theta_{n1}\theta_{ne}}{\theta_{(m-1)e}} \tag{3-47}$$

Return to step 3 above, re-calculate $\theta_{(n+1)e}$, $\theta_{(n+2)e}$, $\cdots$, $\theta_{(m-1)e}$ and $T_{(n+1)e}$, $T_{(n+2)e}$, $\cdots$, $T_{(m-1)e}$., till the new $\theta_{(n+1)e}$, $\theta_{(n+2)e}$, $\cdots$, $\theta_{(m-1)e}$ will satisfy equation (3-38).

5 Then, calculate θ_{me} and T_{me} . Thus, the natural frequencies and mode shapes of a multi-junction, multi-branch torsional vibration system can be determined.

CHAPTER 4

FORCED TORSIONAL VIBRATIONS

This chapter will discuss free vibration with damping, and forced torsional vibration with or without damping.

The analysis of vibration with damping is mainly for studying the influence of the damping to natural frequencies.

The analysis of forced vibration without damping is mainly for studying the amplitude changes of the system under the $M_i = M_{oi} \sin(\omega t + \varphi_i)$.

The analysis of forced vibration with damping is to calculate the largest angular displacements of each disc under the $M_i = M_{oi} \sin(\omega t + \varphi_i)$ and with the effect of the damping.

4.1 An In-line Torsional Vibration System with Damping

Consider the one shaft torsional vibration system with damping. For the generalized condition, any one disc can be described as follows, Figure 4-1.

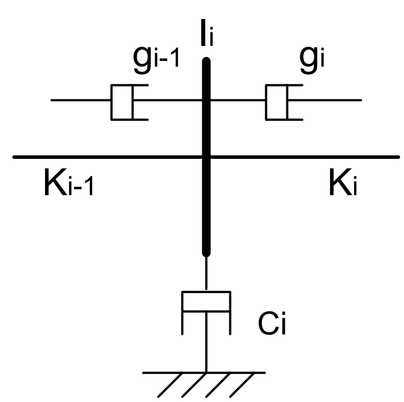

Figure 4 –1 Torsional Vibration Systems with Damping

The damping is assumed to be linear and the system consists of two parts of damping: 1) dashpots between each disc and ground and 2) dashpots between each disc and the one or two part neighbor discs.

The Newton equation of motion for the i th disc (Figure 4-1) is:

$$I_i\ddot{\theta}_i + K_{i-1}(\theta_i - \theta_{i-1}) + K_i(\theta_i - \theta_{i+1}) + c_i\dot{\theta}_i + g_{i-1}(\dot{\theta}_i - \dot{\theta}_{i-1}) + g_i(\dot{\theta}_i - \dot{\theta}_{i+1}) = 0 \quad (4\text{-}1)$$

Where C_i is damping factor for "external" damping (involved with boundary conditions) and g_i is for "internal" damping (between discs).

It is necessary to discuss how to find the end torque that will cause a steady-state forced vibration of an assumed frequency ω with a unit amplitude at the other end. With damping the steady-state motions at the various discs are still harmonic motions at frequency ω, but they no longer have the same phase from disc to disc. That is,

$$\theta_i = \theta_{oi}\sin(\omega t + \psi_i) = A_i \sin\omega t + B_i \cos\omega t \quad (4\text{-}2)$$

Use the shorthand notation $\theta_i = A_i + jB_i$ and substitute equation (4-2) into

equation (4-1):

$$(-I_i\omega^2 + j\omega c_i)\theta_i + (K_i + j\omega g_i)(\theta_i - \theta_{i+1}) + (K_{i-1} + j\omega g_{i-1})(\theta_i - \theta_{i-1}) = 0 \quad (4\text{-}3)$$

This equation differs from the undamped one, Equation 3-20. But, the difference is that $I_i\omega^2$ and K_i are replaced by $(I_i\omega^2 - j\omega c_i)$ and $(K_i + j\omega g_i)$ respectively.

Comparing Equation (4-3) with Equation (3-20), which is for the free vibration without damping, and referring to Equation (3-21)~ (3-23) and Equation (3-24)~(3-27), we can get the following equations:

$$T_1 = (I_1\omega^2 - j\omega c_1)\theta_1 \quad (4\text{-}4)$$

$$T_2 = T_1 + (I_2\omega^2 - j\omega c_2)\theta_2 \quad (4\text{-}5)$$

……

$$T_n = \sum_{i=1}^{n} (I_i \omega^2 - j\omega c_i)\theta_i \tag{4-6}$$

$$\theta_2 = \theta_1 - \frac{T_1}{K_1 + j\omega g_1} \tag{4-7}$$

$$\theta_3 = \theta_2 - \frac{T_2}{K_2 + j\omega g_2} \tag{4-8}$$

$$\theta_4 = \theta_3 - \frac{T_3}{K_3 + j\omega g_3} \tag{4-9}$$

……

$$\theta_n = \theta_{n-1} - \frac{T_{n-1}}{K_{n-1} + j\omega g_{n-1}} \tag{4-10}$$

The computation processes are identical with those of undamped cases (free vibration), except that, as a result of two foregoing replacements, most numerical figures are complex.

In the undamped case, the end torque or "remainder torque" is a real number, which means that the torque is in phase with the motion at the opposite end. Here the remainder torque is a phase with and in quadrate to the motion at the other end. In the undamped case, there are certain frequencies ω for which the remainder becomes zero, which means that the system can have a steady-state vibration without any external excitation. With the presence of damping, this obviously can no longer be the case. The remainder torque never becomes zero. However, for certain values of ω, it becomes a minimum, and we may define these frequencies as the "damped natural frequencies". For small damping, the "damped natural frequencies" differ slightly from the undamped, true, natural frequencies.

4.2 A Multi-Branch Torsional Vibration System with Damping

We have achieved the analysis theory of In-line torsional vibration with damping in section 4.1. Thus, we can also obtain the analysis theory of the multi-branch torsional vibration with damping employing the method of expanding theory from In-line free torsional vibration to multi-branch free torsional vibration.

The method to solve vibration problems for a multi-branch torsional vibration system with damping is similar to the discussions in Section 3.3 (a multi-branch torsional vibration system).

Calculations of the residual torques and deflections for each branch can be obtained by using equations (4-4) ~ (4-10. The compatibility conditions can be satisfied by employing equations (3-28) ~ (3-35). But, in this case, θ_i *and* T_i in these equations are complex numbers.

1 At the junction point, the deflection at each branch is identical.

$$\theta_{1e} = \theta_{2e} = \theta_{3e} = \cdots = \theta_{(n-1)e} \tag{4-11}$$

where θ_{ie} expresses the deflection at the end point of branch i. It is a complex quantity, and it can be written as:

$$\theta_{ie} = \theta_{ie}^{(R)} + j\theta_{ie}^{(I)} \tag{4-11a}$$

Where, $\theta_{ie}^{(R)}$ is the real part of θ_{ie} and $\theta_{ie}^{(I)}$ is the imaginary part of θ_{ie}.

$\theta_{me} = \theta_{ne}$ means that $\theta_{me}^{(R)} = \theta_{ne}^{(R)}$ and $\theta_{me}^{(I)} = \theta_{ne}^{(I)}$.

2 The torque summation of the first (n-1) branches at the junction point should be equal to the starting torque at the last branch.

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} = T_{n1} \tag{4-12}$$

where T_{ie} is the torque at the end point of branch i. $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$, equation (4-12) means that:

$$T_{n1}^{(R)} = \sum_{i=1}^{n-1} T_{ie}^{(R)} \tag{4-13}$$

and

$$T_{n1}^{(I)} = \sum_{i=1}^{n-1} T_{ie}^{(I)} \tag{4-14}$$

3 For the last branch the starting point of calculation is the junction, so that:

$$\theta_{n1} = \theta_{(n-1)e} \tag{4-15}$$

That is, $\theta_{n1}^{(R)} = \theta_{(n-1)e}^{(R)}$, and $\theta_{n1}^{(I)} = \theta_{(n-1)e}^{(I)}$.

4.3 A Multi-Junction, Multi-Branch Torsional Vibration System with Damping

Expanding the multi-branch torsional vibration further, a multi-junction torsioal vibration system can be expressed and furthermore, an analysis theory of multi-junction torsional vibration can be obtained. Consequently, there are several junctions in the systems, in these cases, the calculating process is more complex than the multi-branch torsional vibration.

The method to solve vibration problems for a multi-junction, multi-branch torsional vibration system with damping is similar to the discussions in the Section 3.4 (a multi-branch torsional vibration system).

Calculations of the residual torques and deflections for each branch can be obtained by using equations (4-4) ~ (4-10). Using the same method discussed in Section 4.2 (a Multi-Branch Torsional Vibration System With Damping), the compatibility conditions at the first junction can be satisfied by employing equations (4-11) ~ (4-15).

To satisfy the compatibility conditions at the second junction or next junctions, we can use equations (3-38) ~ (3-47). But, in this case, θ_i *and* T_i in these equations are complex numbers.

1 At the second junction point, deflections at each branch are identical.

$$\theta_{ne} = \theta_{(n+1)e} = \theta_{(n+2)e} = \cdots = \theta_{(m-1)e} \tag{4-16}$$

where θ_{je} expresses the deflection at the end point of branch j (j = n, n+1, n = 2, … , m-1). $\theta_{je} = \theta_{je}^{(R)} + j\theta_{je}^{(I)}$.

In this case, θ_{je} is a complex quantity. $\theta_{me} = \theta_{ne}$ means that $\theta_{me}^{(R)} = \theta_{ne}^{(R)}$ and $\theta_{me}^{(I)} = \theta_{ne}^{(I)}$, ($\theta_{me} = \theta_{me}^{(R)} + j\theta_{me}^{(I)}$; $\theta_{ne} = \theta_{ne}^{(R)} + j\theta_{ne}^{(I)}$).

For the last branch (branch m), the starting point of calculation is the second junction, so that:

$$\theta_{m1} = \theta_{(m-1)e} \tag{4-17}$$

2 The torque summation of the first (n-1) branches at the junction point should be equal to the starting torque at the last branch.

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} = T_{n1} \tag{4-18}$$

where T_{ie} is the torque at the end point of branch i. $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$, equation (4-18) means that:

$$T_{n1}^{(R)} = \sum_{i=1}^{n-1} T_{ie}^{(R)} \tag{4-18}$$

and

$$T_{n1}^{(I)} = \sum_{i=1}^{n-1} T_{ie}^{(I)} \tag{4-19}$$

3 The torque summation of branches from n to (m-1) at the second junction point should be equal to the starting torque at the last branch (branch m), i.e:

$$T_{ne} + T_{(n+1)e} + T_{(n+2)e} + \cdots + T_{(m-1)e} = T_{m1} \tag{4-20}$$

where T_{je} is the torque at the end point of branch j (j = n, n+1, n+2,…, m-1). $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$. Equation (4-20) means that:

$$T_{n1}^{(R)} = \sum_{i=1}^{n-1} T_{ie}^{(R)} \tag{4-21}$$

and

$$T_{n1}^{(I)} = \sum_{i=1}^{n-1} T_{ie}^{(I)} \tag{4-22}$$

4.4 Forced Torsional Vibration of an In-line System with Damping

Consider the forced torsional vibration system with damping, shown in Figure 4-2. In this case, the system will vibrate at frequency of the external force.

For the forced torsional vibration, the main study is to calculate the largest angular displacements of each disc under the $M_i = M_{oi}\sin(\omega t + \varphi_i)$, i = 1, 2, 3,……,n.

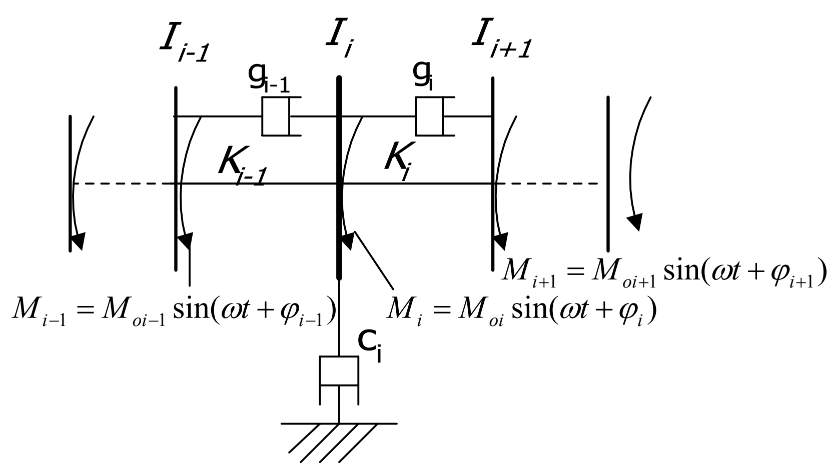

Figure 4-2 Forced Torsional Vibration of an In-line System with Damping

1 Analysis theory of a forced torsional vibration system

For an equivalent torsional vibration system with n discs，assuming the frequency of external forced torque is ω. Then the Newton equation of motion for the *i* th disc (Figure 4-3) will be:

$$
\begin{aligned}
&I_i\ddot{\theta}_i + K_{i-1}(\theta_i - \theta_{i-1}) + K_i(\theta_i - \theta_{i-1}) + c_i\dot{\theta}_i + g_{i-1}(\dot{\theta}_i - \dot{\theta}_{i-1}) + g_i(\dot{\theta}_i - \dot{\theta}_{i+1}) \\
&= M_{oi}e^{j(\omega t+\varphi_i)} = M_i e^{j\omega t}
\end{aligned}
\tag{4-23}
$$

where θ_i is the deflection of the *i* th disc; C_i is damping factor for "external" damping (involved with boundary conditions) and g_i for "internal" damping (between disc i and disc i+1); M_i, φ_i are the complex number amplitude and the initial phase angle of the applied outer torque acted on the *i* th disc.

According to the D'Alembert's Principle，at any time, the torque summation of the applied outer torque、inertia torque and other outer torque of vibration system is always zero. then:

$$
T_n = \sum_{i=1}^{n}[(\omega^2 I_i - i\omega c_i)\theta_i + M_i] = 0 \tag{4-24}
$$

Equation (4 - 24) is the residual torque equation of system. We will discuss a method to solute the equation (4-24).

Let the solution of equation (4-23) be:

$$
\theta_i = \theta_{oi}e^{j(\omega t+\psi_i)} = \theta_i e^{j\omega t} \tag{4-25}
$$

Where ψ_i, θ_i are the initial phase angle and the complex number amplitude of the deflection angle of the ith disc. Substitute equation (4-25) into equation (4-23), and let i =1, 2,⋯,n, we can get the following equations:

$$[(\omega^2 I_1 - j\omega c_1)\theta_1 + M_1] + (K_1 + j\omega g_1)(\theta_1 - \theta_2) = 0$$

$$[(\omega^2 I_2 - j\omega c_2)\theta_2 + M_2] + (K_1 + j\omega g_1)(\theta_2 - \theta_1) + (K_2 + j\omega g_2)(\theta_2 - \theta_3) = 0$$

……

$$[(\omega^2 I_i - j\omega c_i)\theta_i + M_i] + (K_{i-1} + j\omega g_{i-1i})(\theta_i - \theta_{i-1}) + (K_i + j\omega g_i)(\theta_{i-1} - \theta_i) = 0$$

……

$$[(\omega^2 I_n - j\omega c_n)\theta_n + M_n] + (K_{n-1} + j\omega g_{n-1})(\theta_n - \theta_{n-1}) = 0$$

(4-26)

The torque of each disc can be determined as follows:

$$T_1 = (I_1\omega^2 - j\omega c_1)\theta_1 + M_1 \tag{4-27}$$

$$T_2 = T_1 + (I_2\omega^2 - j\omega c_2)\theta_2 + M_2 \tag{4-28}$$

……

$$T_n = \sum_{i=1}^{n}[(I_i\omega^2 - j\omega c_i)\theta_i + M_i] \tag{4-29}$$

By substituting equations (4-27) ~ (4-29) into (4-26), the latter can be arranged as:

$$\theta_2 = \theta_1 - \frac{T_1}{K_1 + j\omega g_1} \tag{4-30}$$

$$\theta_3 = \theta_2 - \frac{T_2}{K_2 + j\omega g_2} \tag{4-31}$$

$$\theta_4 = \theta_3 - \frac{T_3}{K_3 + j\omega g_3} \tag{4-32}$$

……

$$\theta_n = \theta_{n-1} - \frac{T_{n-1}}{K_{n-1} + j\omega g_{n-1}} \tag{4-33}$$

2 Method of numeral calculation

According to Equation (4-27) to (4-33), it can be completed to calculate the amplitude and the torque of each disc of the forced torsional vibration system. But, in order to calculate them practically, we must obtain the θ_1 (a complex number) first.

Because the system is in linear, according to equation (4-29), the residual torque of the system, T_n, can be described as follows:

$$T_n = (a_r + jb_r)\theta_{r1} + (a_i + jb_i)\theta_{i1} + (c_m + jd_m) = 0 \quad (4\text{-}34)$$

Where θ_{r1}, θ_{i1} express respectively the real part and the imaginary part of θ_1; c_m, d_m are the real part and imaginary part of the external torque on the disc i.

Hence we can precisely calculate the value of θ_1 by the following three steps:

1) Let $\theta_1 = 1.0 + j0.0$, without applied outer torque, there will be no item of $M_i \quad (i = 1,2,3,...,n)$ in the equation (4-27) to (4-29). To calculate the residual torque T_n by equation (4-27) to (4-33), we can get $T_n = a_r + jb_r$. Thus, we can obtain a_r, b_r.

2) Let $\theta_1 = 0.0 + j1.0$, without applied outer torque, there will be no item of $M_i \quad (i = 1,2,3,...,n)$ in the equation (4-27) to (4-29). To calculate the residual torque T_n by equation (4-27) to (4-33), we can get $T_n = a_i + jb_i$. Thus, we can obtain.

3) Let $\theta_1 = 0.0 + j0.0$, with outer forced torque, to calculate the residual torque T_n by equation (4-27) to (4-34), we can get $T_n = c_m + jd_m$ and can obtain.

Now, we get a_r, b_r, a_i, b_i, c_m, d_m.

According to Equations (4-24) and (4-34), we can have the following:

$$(a_r + jb_r)\theta_{r1} + (a_i + jb_i)\theta_{i1} + (c_m + jd_m) = 0 \quad \text{(4-34a)}$$

Thus, we solve Equation (4-34a) and obtain:

$$\theta_1 = \theta_{r1} + j\theta_{i1} \quad \text{(4-34b)}$$

$$|\theta_1| = \sqrt{\theta_{r1}^2 + \theta_{i1}^2} \quad \text{(4-34c)}$$

and

$$tg\psi_1 = \frac{\theta_{i1}}{\theta_{r1}} \quad \text{(4-34d)}$$

According to the obtained values θ_{r1} , θ_{i1}, we can calculate the amplitude and the residual torque of all of the mass points (discs) by using Equations (4-27) ~ (4-33), as shown in the following expressions:

$$\begin{aligned} \theta_i &= \theta_{ri} + j\theta_{ii} \\ T_i &= T_{ri} + jT_{ii} \end{aligned} \quad \text{(4-34e)}$$

Therefore, the amplitude and the phase angle of the ith disc can be calculated by the following equations.

$$\begin{aligned} |\theta_i| &= \sqrt{\theta_{ri}^2 + \theta_{ii}^2} \\ tg\psi_i &= \frac{\theta_{ii}}{\theta_{ri}} \end{aligned} \quad \text{(4-34f)}$$

The torsional deflection of the shaft between ith disc and (i+1) th disc is:

$$|\theta_i - \theta_{i+1}| = \sqrt{(\theta_{ri} - \theta_{ri+1})^2 + (\theta_{ii} - \theta_{iri+1})^2} \quad \text{(4-34g)}$$

The torsional stress of the shaft between ith disc and (i+1) disc is:

$$\tau_i = \frac{K_i|\theta_i - \theta_{i+1}|}{0.2d_i^3} \quad \text{(4-34h)}$$

where, d_i is the diameter of the shaft between ith disc and (i+1) disc.

4.5 Forced Torsional Vibration of a Multi-Branch System with Damping

Based on the analysis of section 4.4 (forced torsional vibration of an in-line system with damping), we can obtain the calculating process of a multi-branch torsional vibration system.

The torque and deflection on every disc of each shaft can be calculated by equations (4-27) ~ (4-34) in section 4.4.

The compatibility conditions are similar to section 4.2 (a multi-branch torsional vibration system with damping). The only difference is that there is the effect of the outer torque on each disc, and deflections and the torques are complex numbers.

Now, we discuss the compatibility conditions at the junction.

1 At the junction, the deflections at each branch are identical, that is:

$$\theta_{1e} = \theta_{2e} = \theta_{3e} = \cdots = \theta_{(n-1)e} \quad (4\text{-}35)$$

Where θ_{ie} expresses the deflection at the end point of Branch i. θ_{ie} is a complex quantity, can be written as $\theta_{ie} = \theta_{ie}^{(R)} + j\theta_{ie}^{(I)}$.

$\theta_{me} = \theta_{ne}$ means that $\theta_{me}^{(R)} = \theta_{ne}^{(R)}$ and $\theta_{me}^{(I)} = \theta_{ne}^{(I)}$.

2 The torque summation of the first (n-1) branches at the junction point should be equal to the starting torque at the last branch.

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} + M_{n1} = T_{n1} \quad (4\text{-}36)$$

where T_{ie} is the torque at the end disc of branch i (i=1, 2, 3, ……, n). $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$, M_{n1} is the external torque which act on the first disc of branch n.

Equation (4-34) means that:

$$T_{n1}^{(R)} = \sum_{i=1}^{n-1} T_{ie}^{(R)} + M_{n1}^{(R)} \quad (4\text{-}37)$$

and

$$T_{n1}^{(I)} = \sum_{i=1}^{n-1} T_{ie}^{(I)} + M_{n1}^{(I)} \quad (4\text{-}38)$$

3 For the last branch the starting disc of calculation is at the junction, so that:

$$\theta_{n1} = \theta_{(n-1)e} \quad (4\text{-}39)$$

That is, $\theta_{n1}^{(R)} = \theta_{(n-1)e}^{(R)}$, and $\theta_{n1}^{(I)} = \theta_{(n-1)e}^{(I)}$.

4.6 Forced Torsional Vibration of a Multi-Junction Multi-Branch System with Damping

We have discussed the method and procedure of damped forced tosional vibration of an in-line system and a multi-branch system. Those method and procedures can be expanded to a multi-junction multi-branch torsional vibration system to get solutions.

The torque and deflection on every disc of each shaft can be calculated by equations (4-27) ~ (4-34) described in Section 4.4 (forced torsional vibration of an in-line system with damping).

The compatibility conditions at the first junction are the same as equations (4-35) ~ (4-39) described in section 4.5 (forced torsional vibration of a multi-branch system with damping).

Now, we discuss the compatibility conditions at the second or next junctions.

1 At the junction point, the deflection at each branch is identical.

$$\theta_{ne} = \theta_{(n+1)e} = \theta_{(n+2)e} = \cdots = \theta_{(m-1)e} \quad (4\text{-}40)$$

Where θ_{je} expresses the deflection at the end disc of branch j (j = n, n+1, n=2,…, m-1).

Here, θ_{je} is a complex quantity, and can be written as $\theta_{je} = \theta_{je}^{(R)} + j\theta_{je}^{(I)}$.

$\theta_{me} = \theta_{ne}$ means that $\theta_{me}^{(R)} = \theta_{ne}^{(R)}$ and $\theta_{me}^{(I)} = \theta_{ne}^{(I)}$.

For the last branch (branch m), the starting point of calculation is the second junction, so that:

$$\theta_{m1} = \theta_{(m-1)e} \tag{4-41}$$

2 The torque summation of all the (n-1) branches at the junction point should be equal to the starting torque at the last branch.

$$T_{1e} + T_{2e} + T_{3e} + \cdots + T_{(n-1)e} + M_{n1} = T_{n1} \tag{4-42}$$

where T_{ie} is the torque at the end point of branch i. $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$, M_{n1} is the external torque which acted on the first point of branch n. Equation (4-42) means that:

$$T_{n1}^{(R)} = \sum_{i=1}^{n-1} T_{ie}^{(R)} + M_{n1}^{(R)} \tag{4-43}$$

and

$$T_{n1}^{(I)} = \sum_{i=1}^{n-1} T_{ie}^{(I)} + M_{n1}^{(I)} \tag{4-44}$$

3 The torque summation of branches from n to (m-1) at the second junction point should be equal to the starting torque at the last branch (branch m), i.e:

$$T_{ne} + T_{(n+1)e} + T_{(n+2)e} + \cdots + T_{(m-1)e} + M_{m1} = T_{m1} \tag{4-45}$$

where T_{je} is the torque at the end point of branch j (j = n, n+1, n+2, …, m-1). $T_{ie} = T_{ie}^{(R)} + jT_{ie}^{(I)}$. Equation (4-45) means that:

$$T_{m1}^{(R)} = \sum_{i=n}^{m-1} T_{ie}^{(R)} + M_{m1}^{(R)} \tag{4-46}$$

and

$$T_{m1}^{(I)} = \sum_{i=n}^{m-1} T_{ie}^{(I)} + M_{m1}^{(I)} \tag{4-47}$$

4.7 Forced Torsional Vibration Without Damping

1 Forced Torsional Vibration of An In-Line System Without Damping

In section 4.4 (forced torsional vibration of an in-line system with damping), the forced torsional vibration with damping has been discussed and the analysis theory has been achieved. If we eliminate the damping factor (include external damping and internal damping) from equations (4-27) ~ (4-33), the solution for forced torsional vibration without damping will be obtained.

In detail, the implementation procedures for a forced torsional vibration of an in-line system without damping are obtained when $j\omega c_i$ and $j\omega g_i$ are removed from equations (4-23) ~ (4-34). The equations for forced torsional vibration of an in-line system without damping are expressed as follows:

$$T_1 = I_1\omega^2\theta_1 + M_1 \tag{4-48}$$

$$T_2 = T_1 + I_2\omega^2\theta_2 + M_2 \tag{4-49}$$

……

$$T_n = \sum_{i=1}^{n} (I_i\omega^2\theta_i + M_i) \tag{4-50}$$

$$\theta_2 = \theta_1 - \frac{T_1}{K_1} \tag{4-51}$$

$$\theta_3 = \theta_2 - \frac{T_2}{K_2} \tag{4-52}$$

$$\theta_4 = \theta_3 - \frac{T_3}{K_3} \tag{4-53}$$

……

$$\theta_n = \theta_{n-1} - \frac{T_{n-1}}{K_{n-1}} \tag{4-54}$$

2 Forced Torsional Vibration of A Multi-Branch System Without Damping

Similar to the multi-branch torsional vibration system with damping, we can achieve the method and procedure of multi-branch torsional vibration without damping according to the method described in section 4.5 (forced torsional vibration of a multi-branch system with damping).

However, the difference of with and without damping has been accounted. Equations (4-48) ~ (4-54) are used to calculate residual torque (T_i) and deflection (θ_i). Equations (4-35) ~ (4-39) are used to calculate compatibility conditions.

3 Forced Torsional Vibration of A Multi-Junction Multi-Branch System Without Damping

Based on the previous discussions , the method and procedure of multi-junction system without damping can be obtained accordingly.

The calculating process is similar to forced torsional vibration of a multi-junction multi-branch system with damping discussed in section 4.6.

The torque and deflection on every disc of each shaft can be calculated by equations (4-48) ~ (4-54).

The compatibility conditions at the first junction can be calculated by equations (4-35) ~ (4-39).

The compatibility conditions at the second and next junctions can be calculated by equations (4-40) ~ (4-47).

CHAPTER 5

DEVELOPMENT OF ANALYSIS SOFTWARE

Current software available for analyzing multi-Junction, multi-branch torsional vibration systems, are only for some specific cases and limited scope of engineering application. Users still need to build mathematical models of each individual vibration system. The programs are considered as only a user's calculator for engineering application, not a powerful toolkit.

The goal of this study is to develop an effective and accurate analysis model and build a generalized user friendly graphic software. In the previous discussions, a theory of forced torsional vibration for multi-junction multi-branch system with or without damping has been developed.

The aim of this study is to develop an analysis method and provide a complete software so that designers can use this powerful tool to solve engineering torsional vibration problems.

Based on the above (previous chapters) developed theory and methods, a generalized analysis software, named as MBTV (a Multi-Branch, multi-junction Torsional Vibration analysis software) has been developed. In this chapter, we will briefly introduce the main features and capabilities of the developed software.

5.1 Features of MBTV (Multi-Branch Torsional Vibration Software)

MBTV is able to calculate natural frequencies and mode shapes of multi-branch torsional vibration systems with one or more junctions. It is also able to analyze forced vibration features of multi-branch, multi-junction torsional vibration systems with damping or without damping.

MBTV is an easy use software and is a very powerful toolkit with a user-friendly graphic interface. Designers need only to input data. The program will complete the rest of work automatically, including building the calculating models and deriving solutions. This software has the following features:

- Automatically identify analytical type and its structure of a multi-branch torsional vibration system.
- Automatically build calculating models of analytical vibration systems, and solve them.
- Automatically output the analysis results in text, curve or graphic, by the designer's demands.
- A user-friendly graphic interface.
- Three output styles for exporting the analysis results.
- MBTV is easy to use. It has elaborate help information. Users can implement a project conveniently with help information.

5.2 Architecture Diagram of MBTV

The MBTV contains three types of structure and four cases in each type. There are total twelve vibration cases to be considered in this software. Figure 5-1 shows the architecture diagram of MBTV. Figure 5-2 shows the workflow diagram of MBTV in which it shows the functions of MBTV program. Figure 5-3 shows a problem solving workflow of MBTV program. In the chart, it expressed the steps that the program solves problems (finding natural frequencies ω_i and mode shapes).

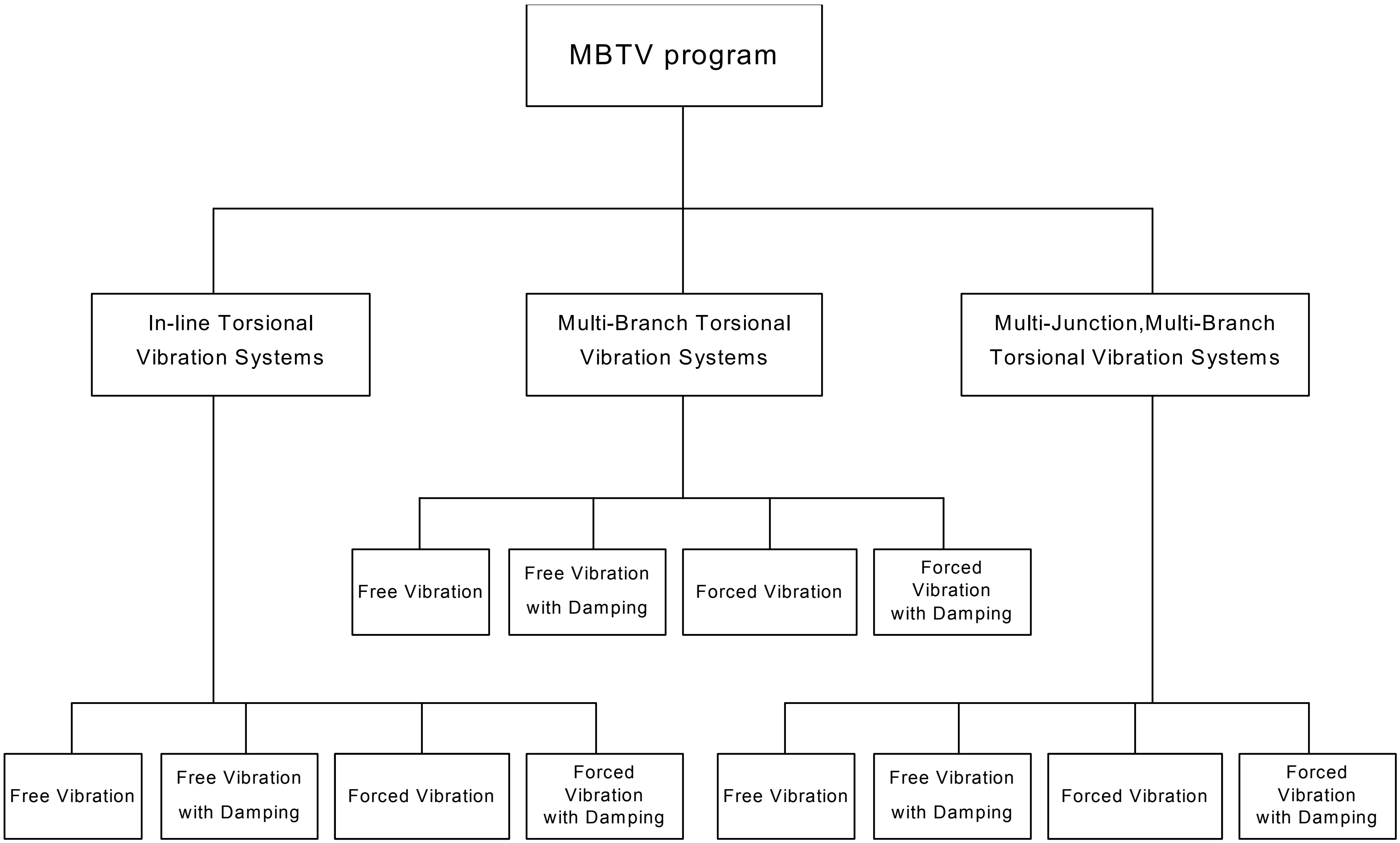

Figure 5-1 Architecture Diagram of MBTV

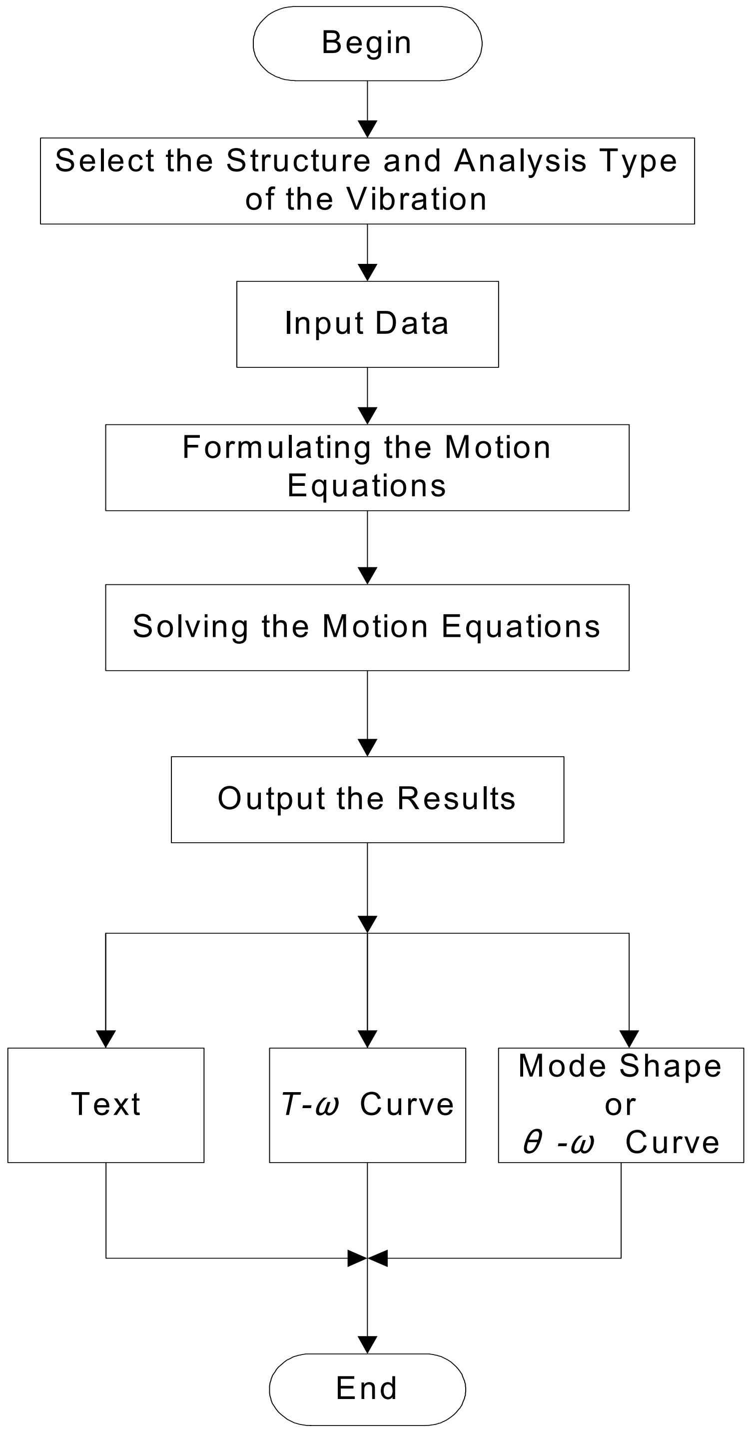

Figure 5-2 Workflow Diagram of MBTV Program

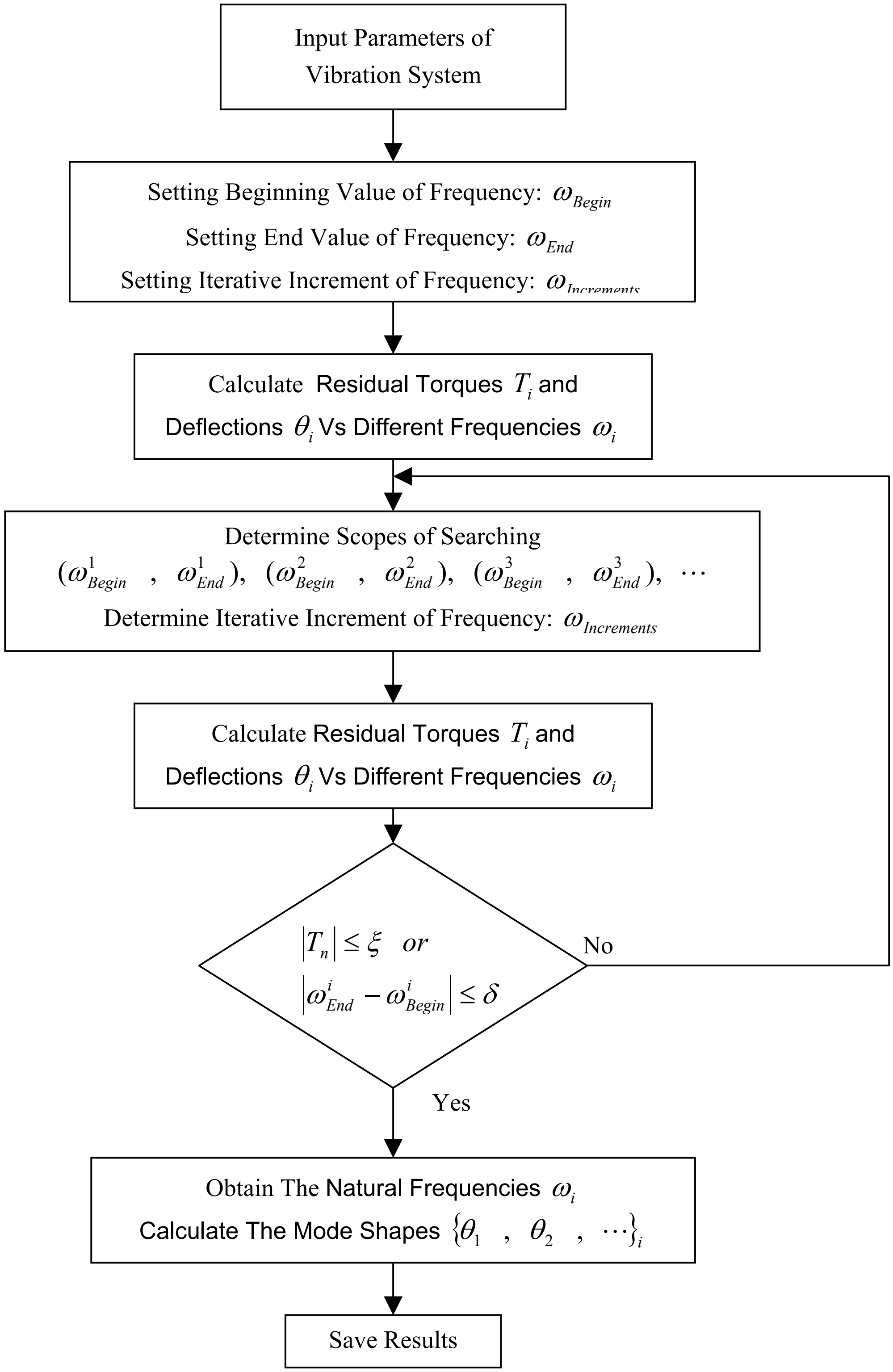

Figure 5-3 Problems Solving Workflow of MBTV Program

Input data will be the number of shafts, number of disks at each shaft, moment of inertia of each disk, stiffness of segment of shafts, and calculating precision as well as any necessary parameters.

The results are of three different cases. For a free vibration system without damping, outputs are natural frequencies and mode shapes of the system in text format, curve or graphic. For a free vibration system with damping, outputs are damped natural frequencies of the system in text format and curve. For a forced torsional vibration system with damping or without damping, outputs are the amplitude and torques at each disc of the system under given external torques in text format.

5.3 Main Function of MBTV Program

The MBTV has been made for solving general torsional vibration problems incurred in industries. In this software, it considered to solve three types of vibration systems. They are: In-line torsional vibration, Multi-branch torsional vibration and Multi-junction multi-branch torsional vibration systems. Each of the three systems includes four kinds of vibration cases. These cases are: Free vibration with or without damping and forced vibration with or without damping. There are twelve vibration cases in total can be solved by the MBTV.

The items that MBTV can calculate are listed as Table 5-1.

Table 5-1 Items of MBTV

Vib-Types / Str-Types	Free Vibration	Free Vibration with Damping	Forced Vibration	Forced Vibration with Damping
In-line	Natural Frequencies Mode Shapes	Damped Natural Frequencies	Amplitude at each disk Torque at each disk	Amplitude at each disk Torque at each disk
Multi-Branch	Natural Frequencies Mode Shapes	Damped Natural Frequencies	Amplitude at each disk Torque at each disk	Amplitude at each disk Torque at each disk
Multi-Junction Multi-Branch	Natural Frequencies Mode Shapes	Damped Natural Frequencies	Amplitude at each disk Torque at each disk	Amplitude at each disk Torque at each disk

In the above table, the amplitude is the angle displacement of each disk. Each disk represents for each gear or mass body on the shafts. Mode shapes mean the angular displacements θ_i corresponding to the natural frequencies.

Except free vibration system, the torques and amplitudes are complex numbers in other vibration cases.

5.4 Programming of MBTV

MBTV program is developed in Visual Basic V6.0, in which the environment is supported by operation systems in Windows 2000 or XP.

MBTV program consists of two modules and 16 forms. The modules are MbCommon and ModPrinting. In MbCommon module, it contains many calculating functions to be used in most forms. In ModPrinting module, it has programs for printing and printing view. There are 16 Forms used in the MBTV, they are: frmFace, frmSelect , frmPreview frmAbout, ildata1, ildata2, ildata3, ildata4, mbdata1, mbdata2, mbdata3, mbdata4, mjdata1, mjdata2, mjdata3 and mjdata4.

(1) Form frmFace is the welcome and instruction interface of MBTV, with File menu and Help menu. In addition, there are three menu commands in this form. i.e. New/ Open/ Exit. Also, there is a message telling users that MBTV can solve three structure types (In-Line system, multi-branch system and multi-junction multi-branch system) and four analysis types (free vibration without damping, free vibration with damping, forced vibration without damping, and forced vibration with damping) for each structure type,.

(2) Form frmSelect is a selecting interface of MBTV for users to select the structure type and analysis type.

(3) Form frmPreview is a print view interface with a scale window and buttons of Print, Zoom in, Zoom out, close, etc.

(4) Form frmAbout is a interface about MBTV copyright.

(5) Form ildata1 is a interface with three windows, several buttons and command menus for free vibration analysis of in-line torsional systems.

(6) Form ildata2 is a interface with three windows, several buttons and command menus for free vibration with damping analysis of in-line torsional systems.

(7) Form ildata3 is a interface with three windows, several buttons and command menus for forced vibration with damping analysis of in-line

torsional systems.

(8) Form ildata4 is a interface with three windows, several buttons and command menus for forced vibration without damping analysis of in-line torsional systems.

(9) Form mbdata1 is a interface with three windows, several buttons and command menus for free vibration analysis of multi-branch torsional systems.

(10) Form mbdata2 is a interface with three windows, several buttons and command menus for free vibration with damping analysis of multi-branch torsional systems.

(11) Form mbdata3 is a interface with three windows, several buttons and command menus, forced vibration with damping analysis of multi-branch torsional systems.

(12) Form mbdata4 is a interface with three windows, several buttons and command menus for forced vibration without damping analysis of multi-branch torsional systems.

(13) Form mjdata1 is a interface with three windows, several buttons and command menus for free vibration analysis of multi-junction, multi-branch torsional systems.

(14) Form mjdata2 is a interface with three windows, several buttons and command menus for free vibration with damping analysis of multi-junction, multi-branch torsional systems.

(15) Form mjdata3 is a interface with three windows, several buttons and

command menus for forced vibration with damping analysis of multi-junction, multi-branch torsional systems.

(16) Form mjdata4. is a interface with three windows, several buttons and command menus for forced vibration without damping analysis of multi-junction, multi-branch torsional systems.

5.5 MBTV Software Structure Diagram

MBTV program structure explains the logistic procedures about how MBTV to solve the torional vibration problems. Calculating procedures are described in steps as shown in Figure 5-4 (a) and 5-4 (b).

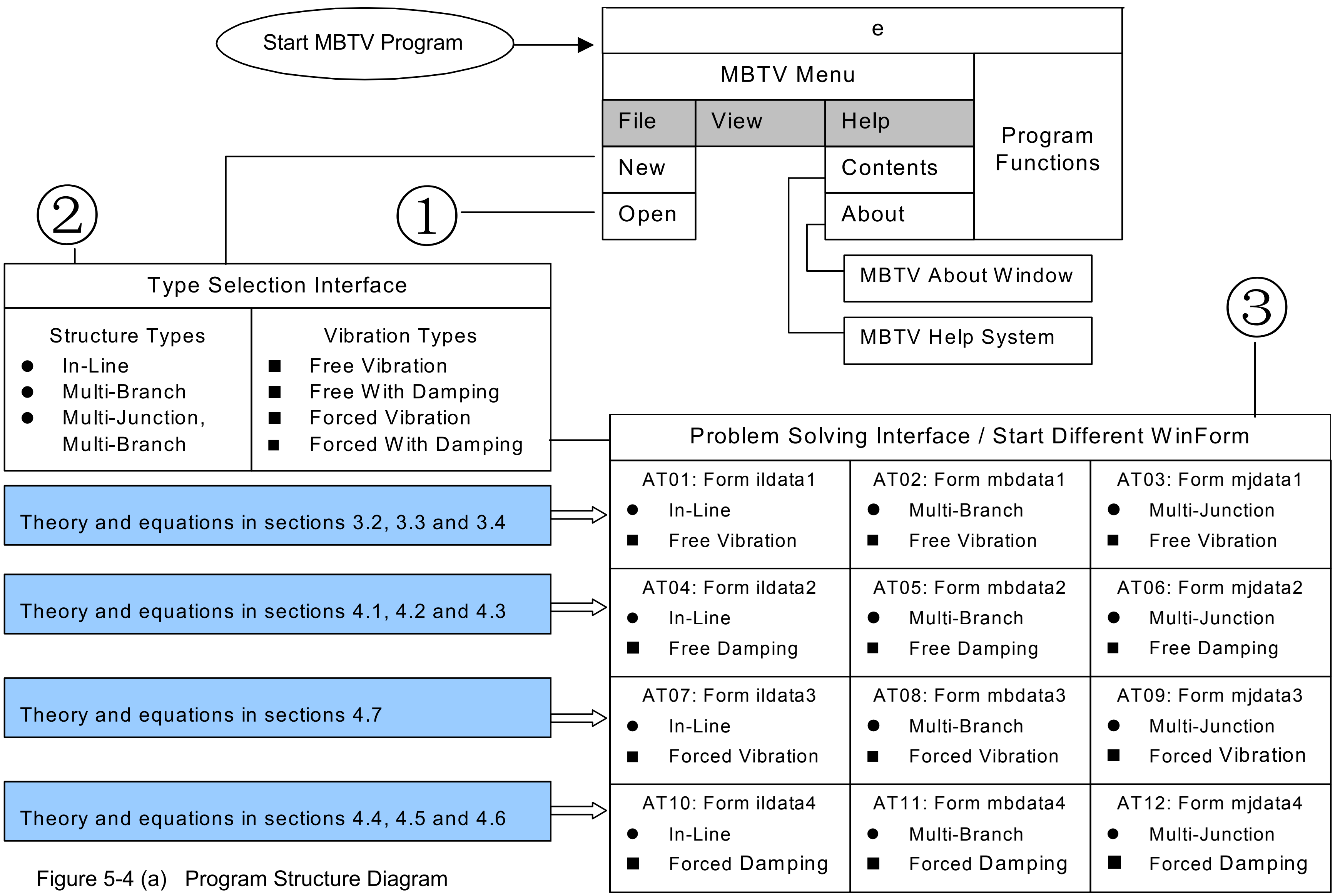

Figure 5-4 (a) Program Structure Diagram

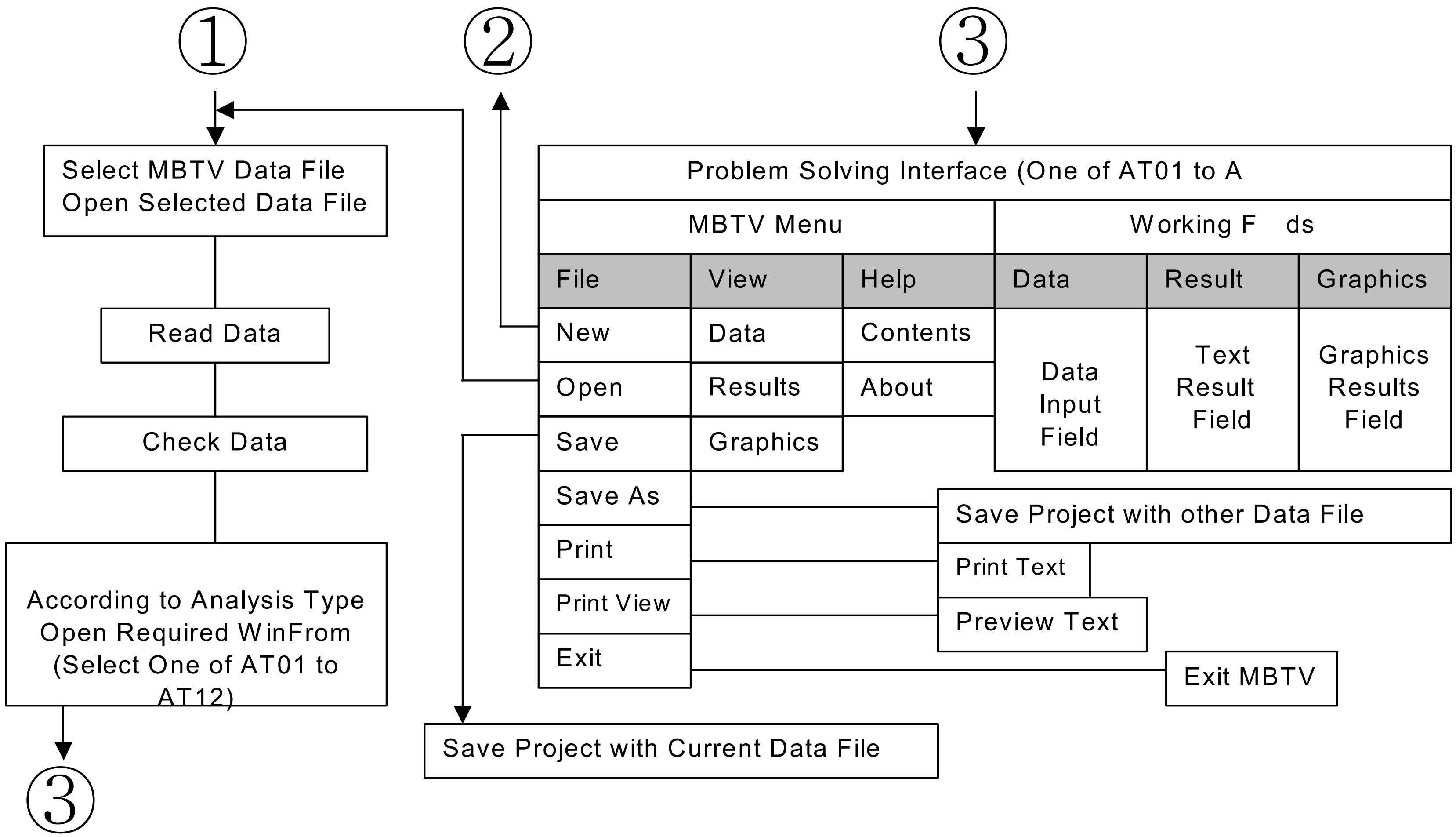

Figure 5-4 (b) Program Structure Diagram

5.6 Interfaces of MBTV Program

The MBTV is a easy use and friendly interfaced software. It provides any necessary interfacing windows for entering data and selections. In this section, it introduces major functions and capabilities of interfaces. These interfaces of MBTV program are shown from Figure 5-5 to Figure 5-16.

5.6.1 Main Interface of MBTV

The MBTV will display a greeting interface on screen after starting the program. On this screen, it demonstrates the main functions of MBTV program as shown in Figure 5-5.

The main interface tells users that MBTV handles three types of analysis structure and four types of vibration. These three types of analysis structure are: (1) In-Line Tortional Vibration Systems, (2) Multi-Branch Tortional Vibration Systems, and (3) Multi-Junction Multi-Branch Torsional systems. The four types of vibration are: (1) Free Vibration without damping, (2) Free Vibration with Damping, (3) Forced Vibration without damping, and (4) Forced Vibration with Damping.

In the Figure 5-5 interface screen, select “New Command” from File Menu, then, we will be able to open the type selection interface as shown in Figure 5-6.

5.6.2 Type Selection Interface of MBTV

Figure 5-6 shows the type selection interface of MBTV. Through this interface, we must select the structure type and analysis type of vibration system to be analyzed by this program. There are three kinds of structures for selection. They are in-line systems, multi-branch systems and multi-junction multi-branch systems respectively. There are four analysis types for selection. They are free vibration without damping, free vibration with damping, forced vibration with damping, and forced vibration without damping respectively.

After selecting structure and analysis type, we can double click “Start” button and goes to “Data Input” interface as shown in Figure 5-7.

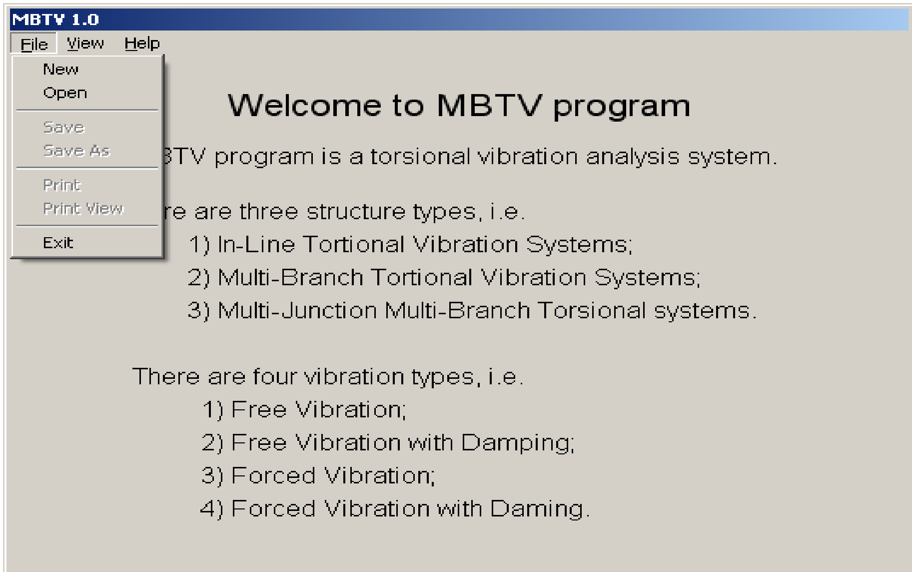

Figure 5-5 MBTV Capabilities from File Menu

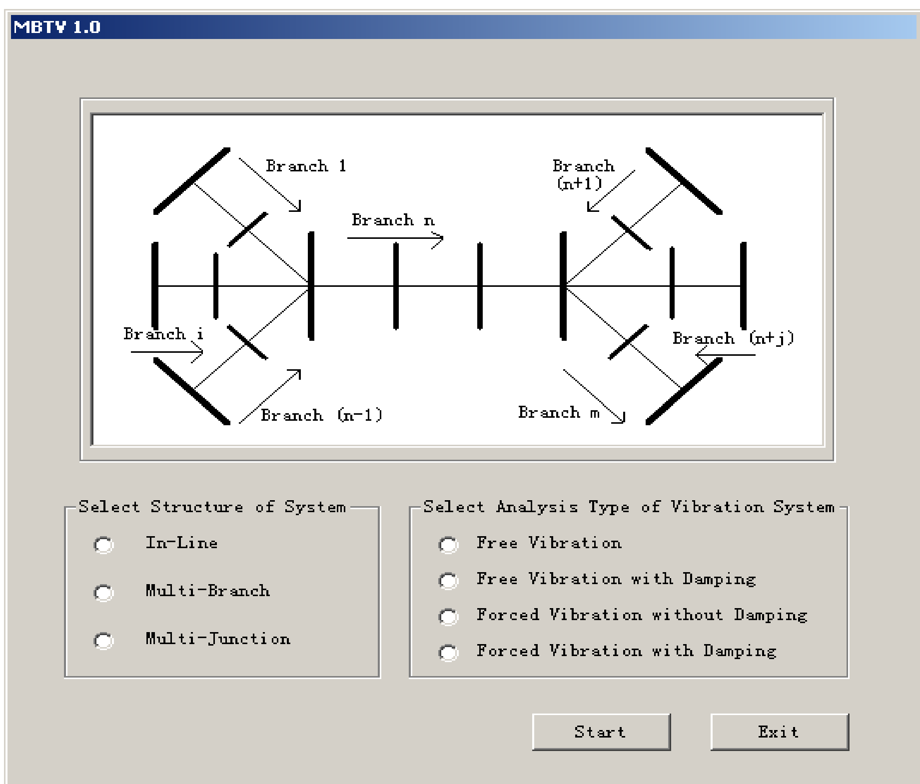

Figure 5-6. Type Selection Interface of MBTV

5.6.3 Data Input Interface

Once "Multi-Branch" is selected from "Select Structure of System" dialog box and "Free Vibration" is selected from "Selection Analysis Type of Vibration system" dialog box in Figure 5-6, we can open the data input interface. It is shown in Figure 5-7.

From this interface selection, we can input the structure data and the system characteristic parameters. There are two methods to input system data: on line interaction to enter in data as shown in figure 5-7 and inputting data by a data file. A data of parameters can be created before running the analysis system.

The content information of this interface varies from cases depending on vibration systems and type selected.

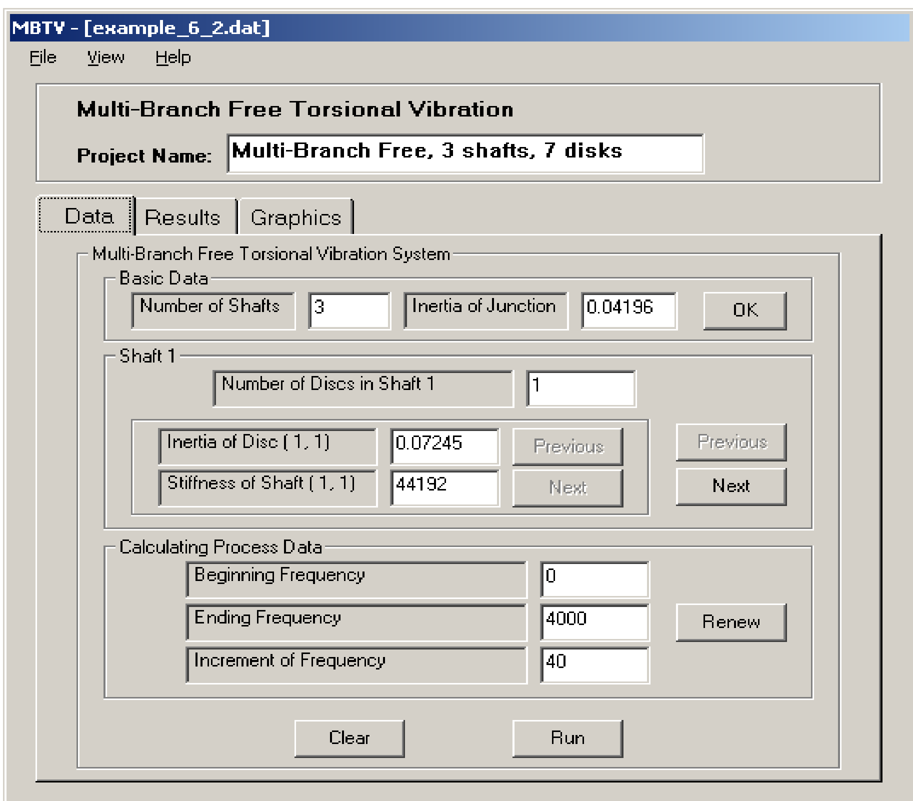

Figure 5-7 Data Input Interface of MBTV

In Figure 5-7, the file name is displayed in the window title column, and the current file name is “Multi_B_Free_01.dat” (representing for multi-branch free vibration, one data). “Project Name” column is used to input the current project name. The buttons of “OK”, “Precious”, “Next” and “Clear” are used to control the data input process.

After inputting the required dada and verified, we can click “Run” button to automatically start the software, process data, and do necessary calculation to get results. Once the above process finished, MBTV automatically changes to “Results” interface window. This interface window lists out the calculated results of natural frequencies in text format as shown in Figure 5-8.

5.6.4 Results Interface

The text results interface of MBTV is shown in Figure 5-8.

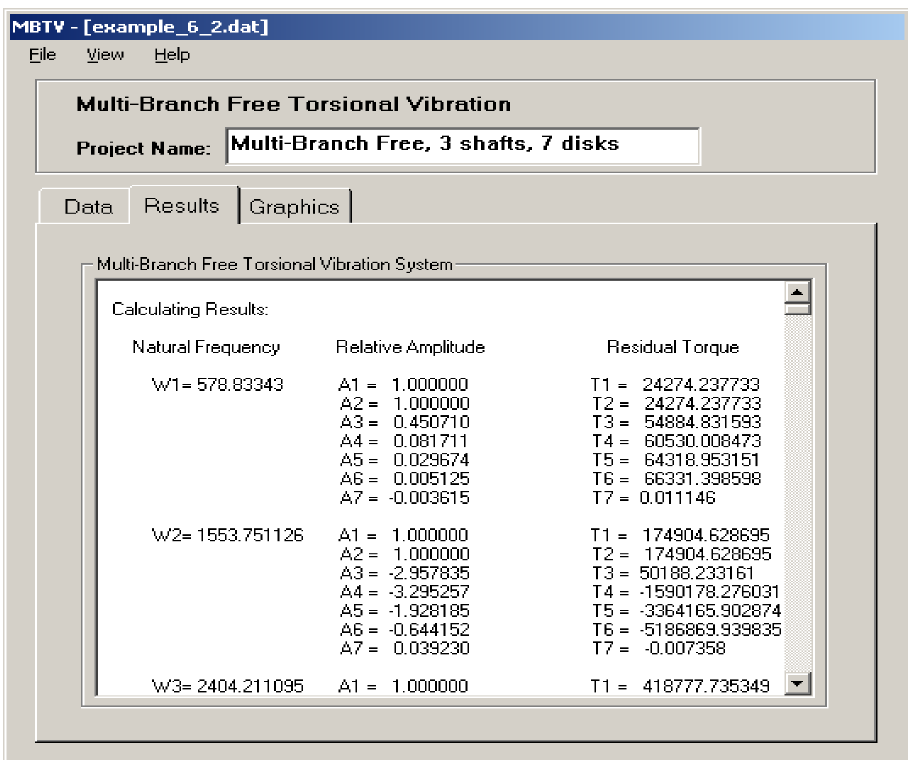

Figure 5-8 Results Interface of MBTV

In this interface, ω_1 represents the first natural frequency, ω_2 represents the second natural frequency and ω_3 represents the third natural frequency, and so on. θ_1, θ_2, θ_3, …, θ_7 represent the mode shapes with corresponding natural frequency.

5.6.5 Graphics Interface

In this interface, once selected the "Graphics" button, "Graphics" window will be displayed. Figure 5-9 shows the graphic results interface of MBTV.

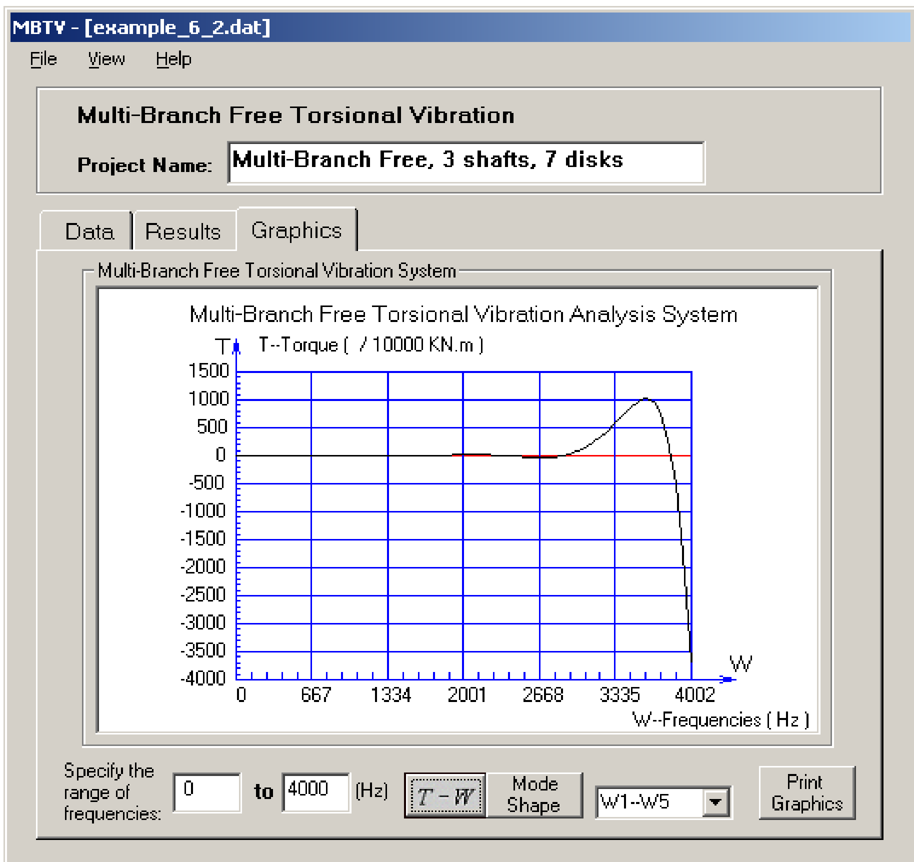

Figure 5-9 Graphics Interface of MBTV

In Figure 5-9, the $T-\omega$ curve shows a relationship of residual torques with frequencies. When a residual torque is equal to zero, the corresponding frequencies are the system natural frequencies to be calculated.

In order to make analyzers easily identify natural frequencies and to display the $T-\omega$ curve more clearly, the MBTV made it possible select a range of frequency first, then, click the "$T-\omega$" button. It will display the selected segment of $T-\omega$ curve in a desired scale.

The following two interfaces demonstrate the features. Figure 5-10 displays a segment of 500 Hz to 1000 Hz for corresponding frequencies. Figure 5-11 displays a segment of 1500 Hz to 2000 Hz for corresponding frequencies. In this way, both of these curves are able to show accurate values of natural frequencies to meet analyzers purpose.

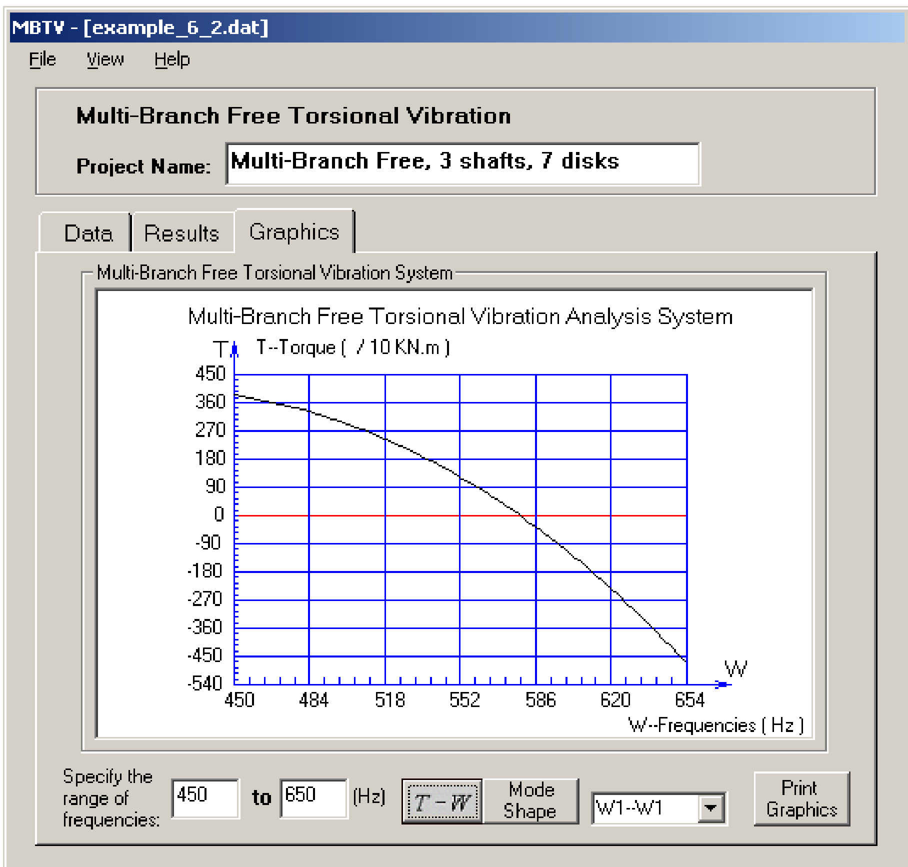

Figure 5-10 $T-\omega$ Curve in a Selected Frequency Range (450 ~ 650 Hz)

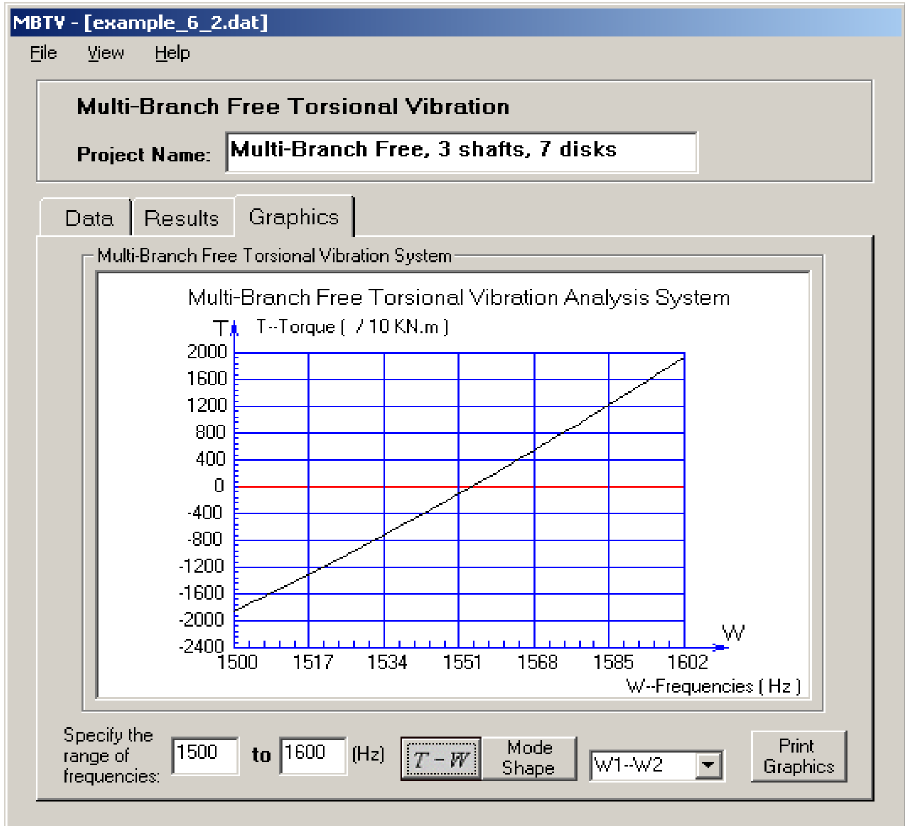

Figure 5-11 $T-\omega$ Curve in a Selected Frequency Range (1500 ~ 1600 Hz)

In any graphic windows of figure 5-9, 5-10 and 5-11, once click "Mode Shape" button, MBTV will display the mode shapes of analyzed vibration system. This feature is shown in Figure 5-12.

MBTV has the function to display every each mode shape with its corresponding natural frequency. It can be down through figures 5-9 to 5-11, by selecting any one particular natural frequency ω from the frequency pull down window on the right of Mode Shape button. Once a frequency is selected, then click "Mode Shape" button, the modem shape will be displayed on screen. Figure 5-13 shows one mode shape at a particular natural frequency.

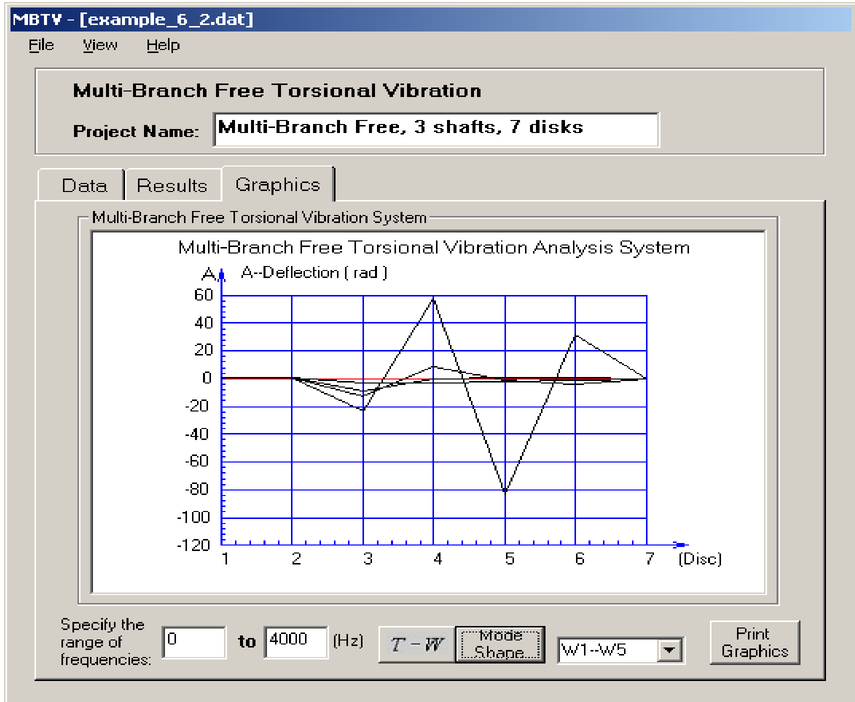

Figure 5-12 Display of Mode Shapes

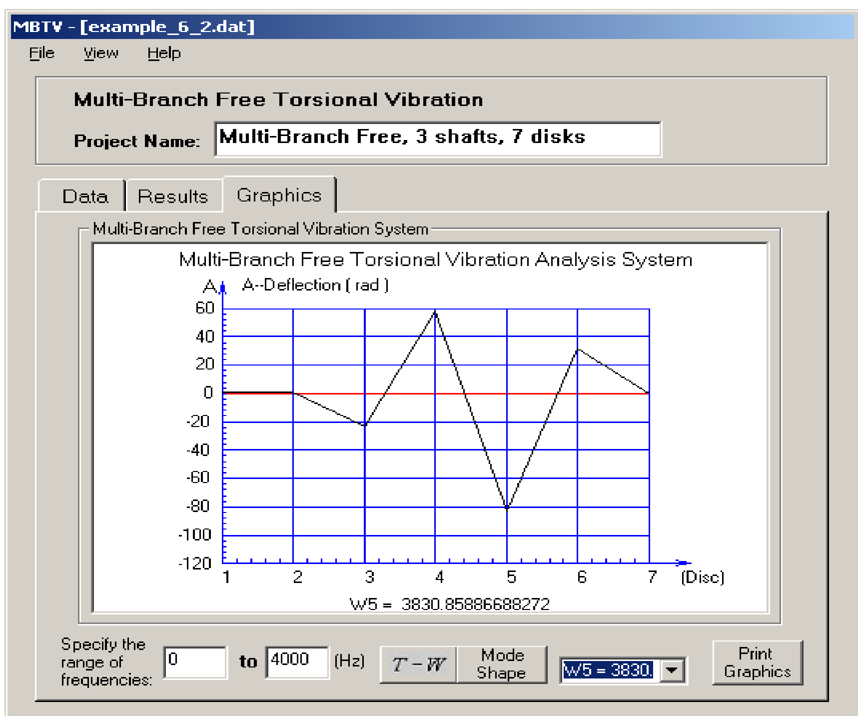

Figure 5-13 Display of One Mode Shape

Printing function is built in the MBTV with a normal Microsoft windows operation. To print out graphic results, we just need simply to select the "Print Graphic" button in any graphic windows shown in figures 5-9 to 5-13. Current graphics ($T-\omega$ Curves or mode shapes) can be printed out as needed by the above operations.

5.7 Main Commands of MBTV Program

MBTV is working under Microsoft windows environment. In general, it is in compliance with all Microsoft windows command applications as introduced in the section 5-4 of this chapter. There are several dedicated commands that apply to the MBTV. The following statements are a brief introduction of those applications.

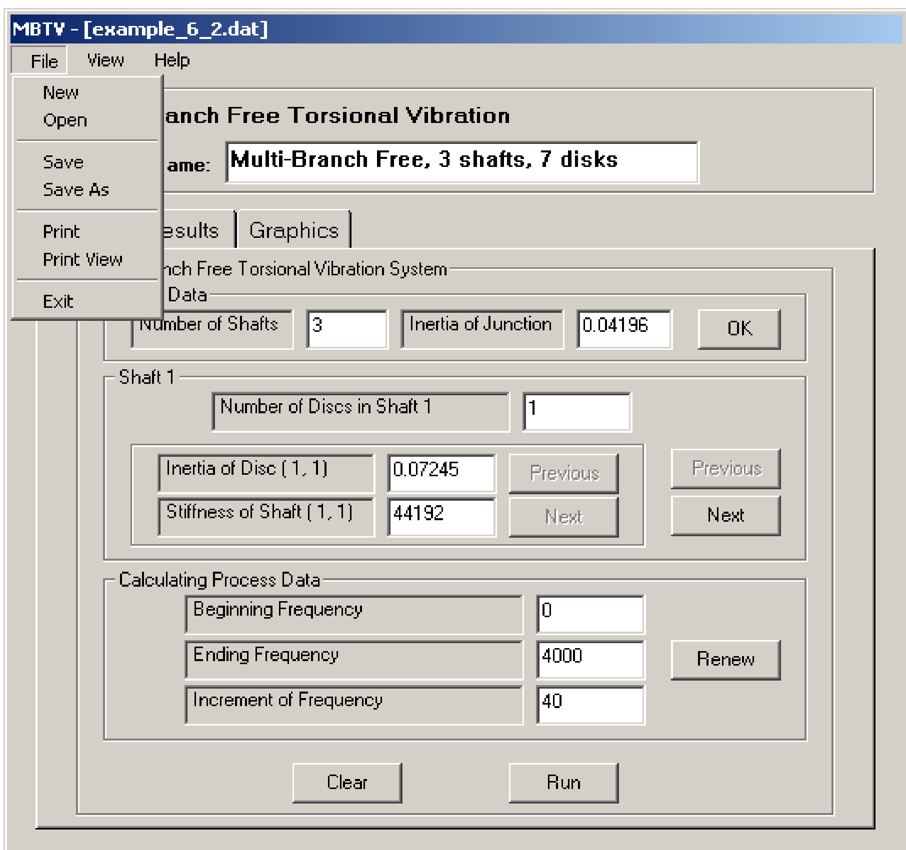

Figure 5-14 File Commands

5.7.1 File Commands

There are seven File-commands in the MBTV. They are New, Open, Save, Save As, Print, Print View and Exit as shown in Figure 5-14.

When we just start MBTV, commands of “Save”, “Save As”, “Print” and “Print View” are not enabled. Because at this stage, there is no any data since a project has not been establish yet (refer to Figure 5-5).

“New” command is used to build a new project. When we select this command, we will open “Type Selection” interface as displayed in Figure 5-6.

“Open” command is used to open an existed project. When we select this command, we will open “File Selection” interface as displayed in Figure 5-15.

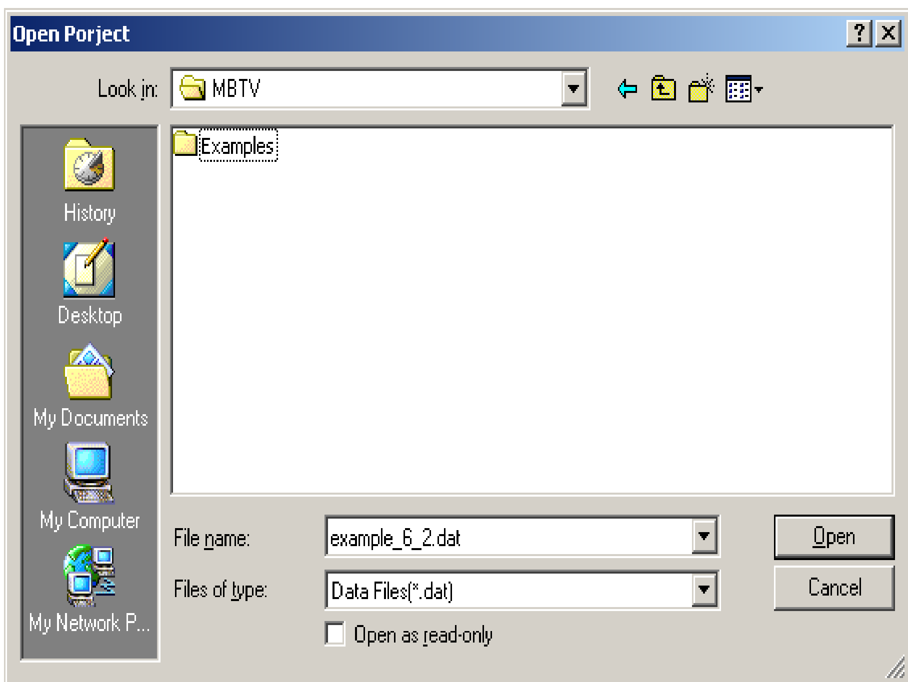

Figure 5-15 Open Command

“Save” command is used to save the current project data file with the current data file name. If the current project data has not been named yet, MBTV will automatically execute “Save As” command.

"Save As" command is used to save the current project data file with a new file name. It is shown in Figure 5-16.

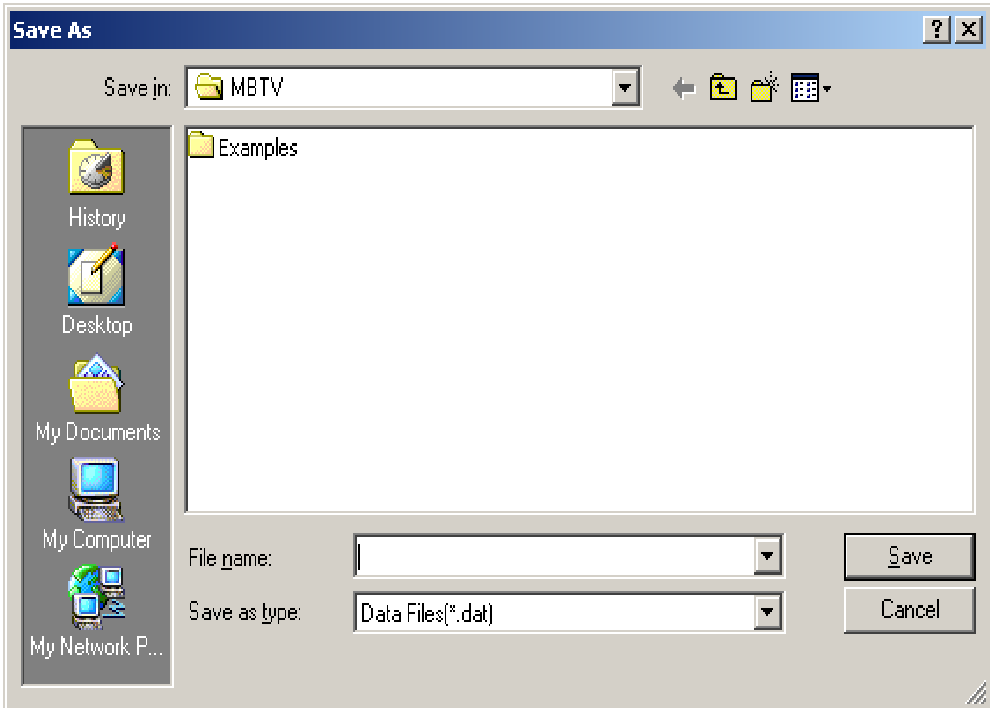

Figure 5-16 Save As Command

"Print" command is used to print out hard copies of the text results or graphics results to printer.

"Print View" command is used to view the text results and the input data before a printing job. This function, shown in Figure 5-17.

For example, In Figure 5-17, n=3 means that there are three shafts in the vibration system. Ij=0.04196 means that the moment of inertia of a junction disc is 0.04196. I(i, j) represent moments of inertia of discs and K(i, j) represent the segment stiffness on the shaft i. ω_1 is the first natural frequency of the system and θ_{01}, $\theta_{02}, \cdots$ are the mode shape at ω_1.

"Exit" command is used to exit MBTV.

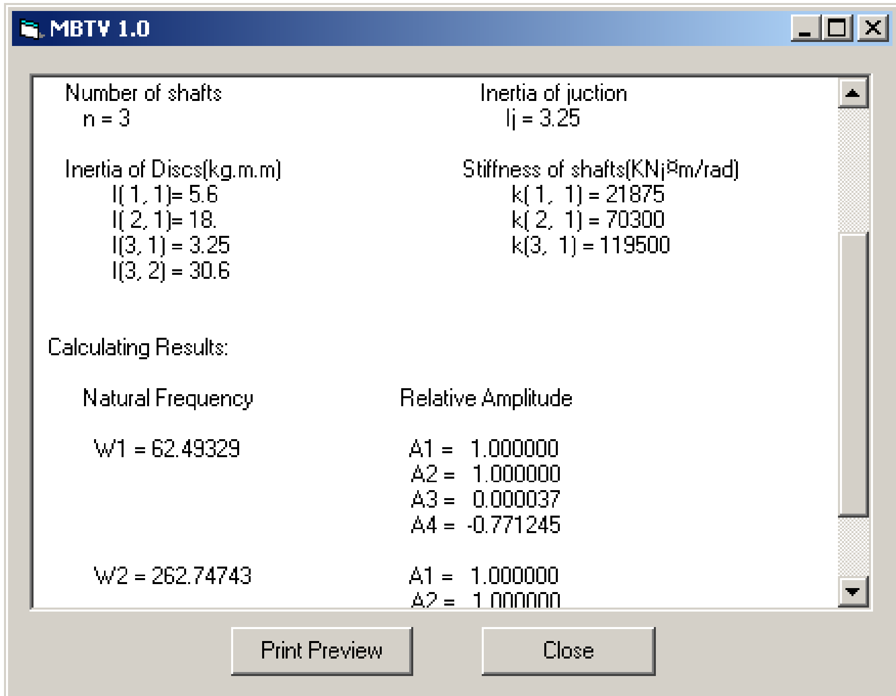

Figure 5-17 Print View Command

5.7.2 View Commands

There are three “View” commands in the MBTV. They are used for changing “Input Data”, “Results” and “Graphics” windows. The View commands interface window is shown in Figure 5-18.

“Data” command is used to change the current window to “input data” window which is shown in Figure 5-7.

“Results” command is used to change the current window to “results” window as shown in Figure 5-8.

“Graphics” command is used to change the current window to “Graphics” window as introduced in Figure 5-9 to Figure 5-12.

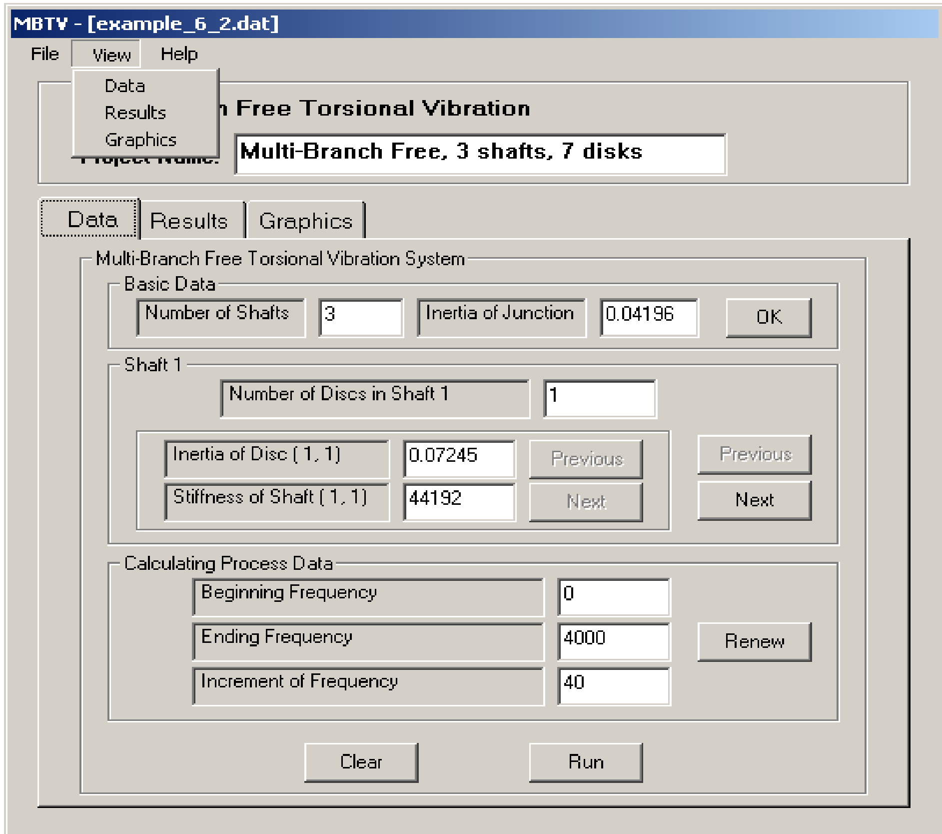

Figure 5-18 View Commands

5.7.3 Help Commands

If a user has problems using the software, one can always click the F1 keystroke on keyboard or open the Help menu to look for help information. The MBTV software has an elaborate screen help information system built in for questions which users may encounter during operation.

There are two "Help" commands which are used to supply the help information for MBTV. They are "Content" and "About".

"Contents" command is used to open the help system of MBTV and tell users how to install and use MBTV. "About" command is used to open the about MBTV. Users can solve problems encountered by looking for certain topics. These two useful help interface windows are shown in Figure 5-19 and Figure 5-20.

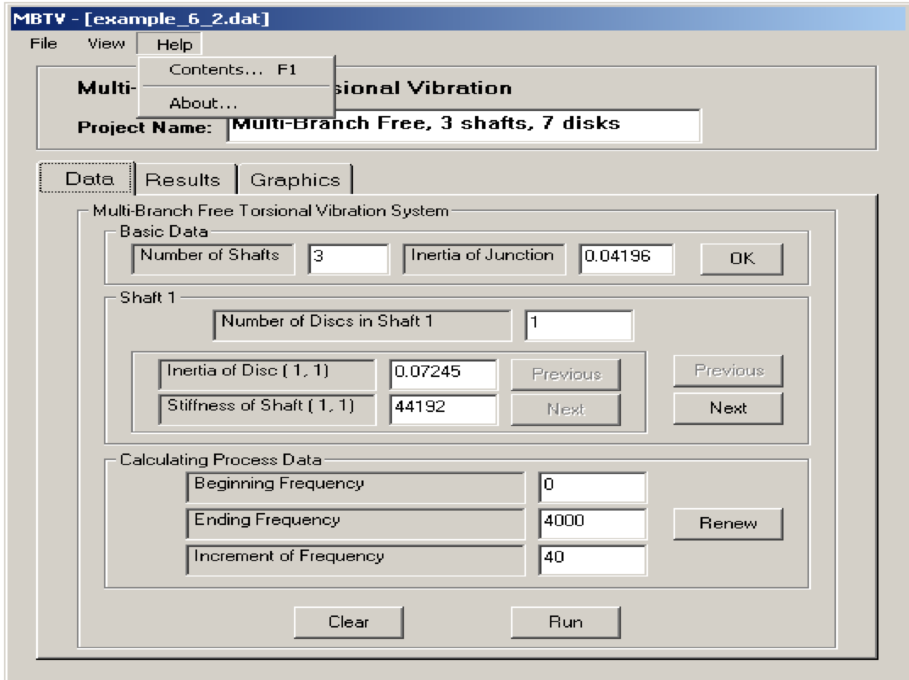

Figure 5-19 Help Commands

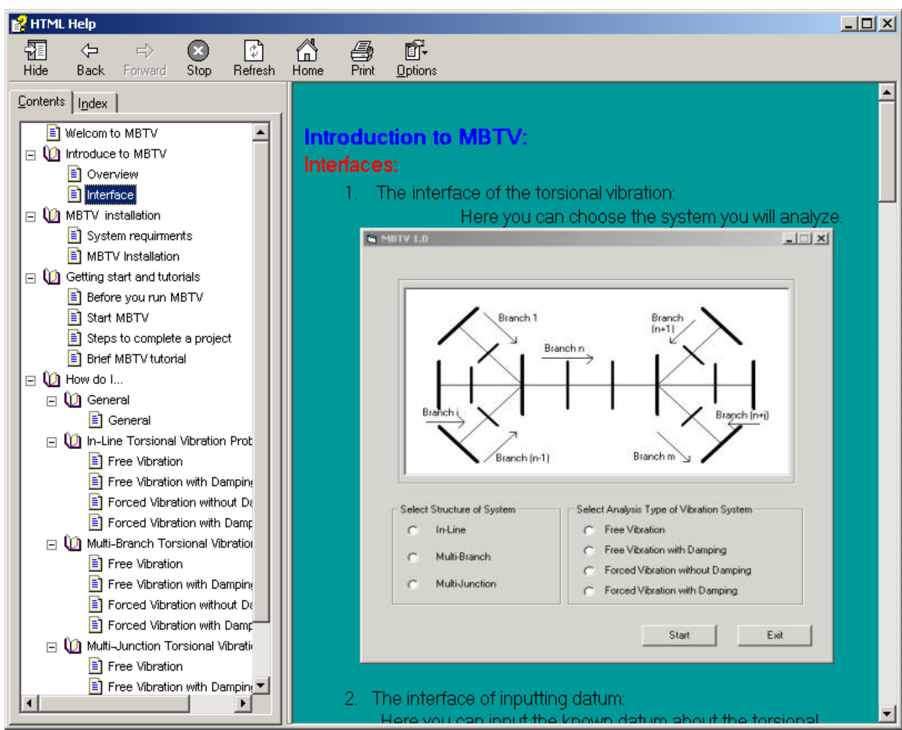

Figure 5-20 Help System of MBTV

5.8 Steps of Using MBTV to Solve Problems

There are seven steps of using MBTV program to solve a torsional vibration problem.

1 Start MBTV Program

From Start menu on the left-down corner of your computer, select “All Programs”/MBTV/MBTV 1.0. Then start MBTV, the main interface of MBTV will have a display as shown in Figure 5-5.

2 Build a New Study Case

There are two methods to build a new study case. One is to select “File/New” command and anther is to select “File/Open” command and “File/Save As” command.

3 Select Analysis Type

After selecting “File/New”, it will open “Type Selection” interface as shown in Figure 5-6. With this window, we can select the required (needed) analysis type.

4 Input Data

In “Data” window as shown in Figure 5-7, input all the structure type data and the characteristic parameters of a system. Using the command buttons in this interface window, we can process and edit data.

5 Calculation and Solution

After input and checked data, we can click “Run” button in “Data” interface window to start the process of calculation and get solutions.

6 View the Results

In this stage, there are two ways to view results. We can use view command in “View” menu for changing windows to check text results or graphics results. We can also select “Data” or “Results” or “Graphics” in the interfaces directly to get results.

7 Print Results

We can use “Print Graphics” command in “Graphics” window to print the graphics results, and use “File/Print” or “File/Print View” commands to print the text results.

The above seven steps are used in MBTV to solve torsional vibration problems. After it is finished, use “File/Save” or “File/Save As” commands to save the results and current project data, then use “File/Exit” commands to exit MBTV.

CHAPTER 6

VERIFICATION OF ENGINEERING EXAMPLES WITH THIS STUDY

We shall use the proposed theory, method and program of this study to solve some practical engineering examples. The simulation results obtained by the program will be compared to those obtained analytically as well as those given in references.

6.1 A Branched-Geared System

A branched-geared system is shown in Figure 6-1. We will use MBTV to calculate its natural frequencies and mode shapes. We will also compare MBTV results with tested results from Reference [42].

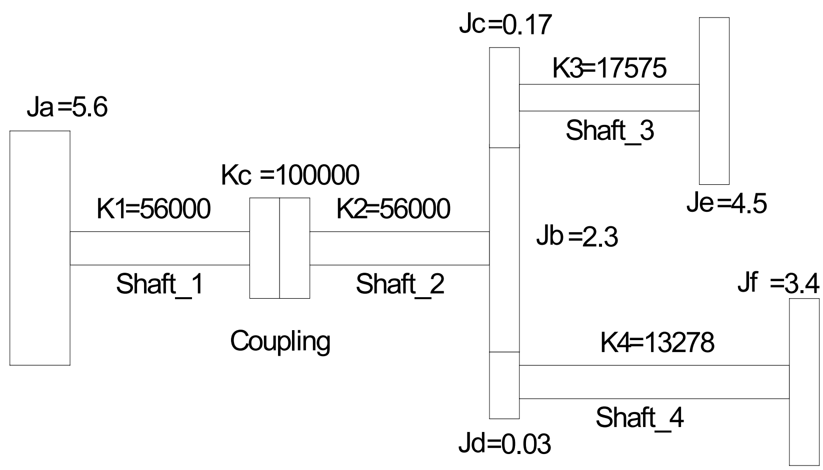

Figure 6-1 A Branched-Geared System

Assume the inertial effect of shafts and coupling is negligible. The gear ratio of these gears are Jb:Jc =1:2 and Jb:Jd =1:3. Data of the system are shown as follows: $Ja = 5.6m^2 Kg$, $Jb = 2.3m^2 Kg$, $Jc = 0.17m^2 Kg$, $Jd = 0.03m^2 Kg$, $Je = 4.5m^2 Kg$, $J_f = 3.4m^2 Kg$, $K_1(Shaft\_1) = 56000 N.m/rad$, $Kc(Coupling) = 100000 N.m/rad$, $K_2(Shaft\_2) = 56000 N.m/rad$,

$K_3(Shaft\_3) = 17575 N.m/rad$, $K_4(Shaft\_4) = 13278 N.m/rad$.

In order to find the natural frequencies, the follows steps are necessary to get its solutions.

■ **Solution Steps**

1) Determine the number of branches and the values of inertia and stiffness in each branch. Set one branch as reference and in put the speed ratio for each branch with regard to the reference branch.

In this work case, take Shaft_1 as a reference branch and the equivalent system can be described as a multi-branch, single junction, free vibration without damping. This system is shown as Figure 6-2.

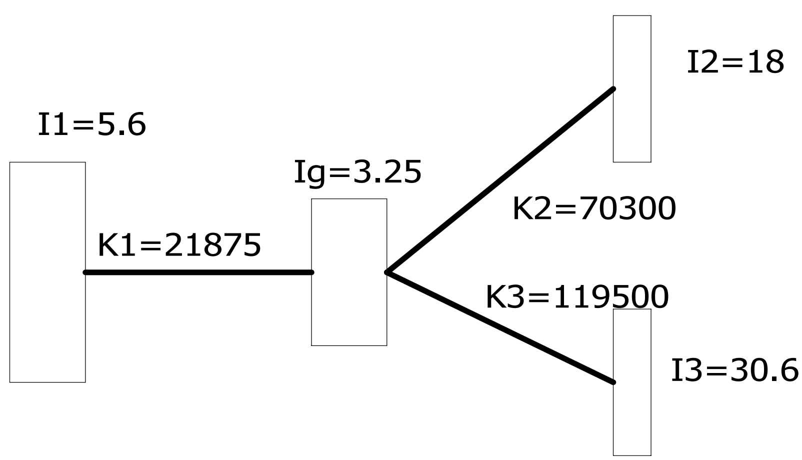

Figure 6-2 Equivalent System

2) Calculate the equivalent values of inertia I_i and stiffness K_i relevant to the reference branch. In the case shown in Figure 6-2, parameters are:

$$I_1 = J_a = 5.6 m^2 Kg$$

$$\frac{1}{K_1} = \frac{1}{K_1(Shaft\_1)} + \frac{1}{K_c(Coupling)} + \frac{1}{K_2(Shaft\_2)} = \frac{1}{56000} + \frac{1}{100000} + \frac{1}{56000}$$

$$K_1 = 21875 N \cdot m / rad$$

$$I_2 = 2 \times 2 \times J_e = 2 \times 2 \times 4.5 = 18 m^2 Kg$$

$$K_2 = 2 \times 2 \times K_2 (Shaft\_3) = 2 \times 2 \times 17575 = 70300 N \cdot m / rad$$

$$I_3 = J_b + 2 \times 2 \times J_c + 3 \times 3 \times J_d = 2.3 + 2 \times 2 \times 0.17 + 3 \times 3 \times 0.03 = 3.25 m^2 Kg$$

$$K_3 = 3 \times 3 \times K_4 (Shaft\_4) = 3 \times 3 \times 13278 = 119502 N \cdot m / rad$$

$$I_4 = 3 \times 3 \times J_f = 3 \times 3 \times 3.4 = 30.6 m^2 Kg$$

3) Run Program MBTV to calculate the values of the residual torques with different frequencies, and obtain the results including natural frequencies and mode shape of this vibration system.

(a) Input the structure data and characteristic parameters, shown in Figure 6-3. Select a proper beginning frequency, ending frequency and increment of frequency, so that the software will start processing calculations and obtain natural frequencies.

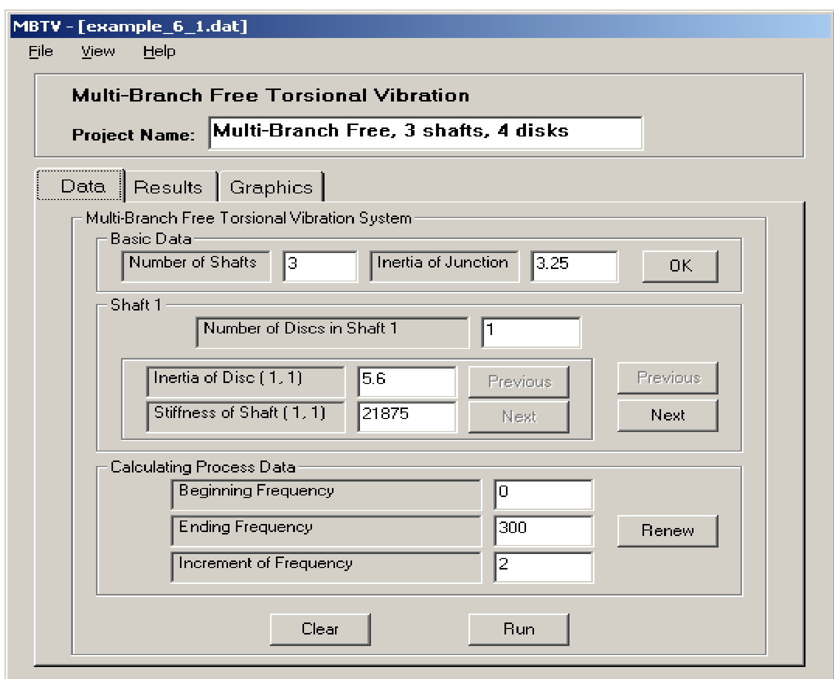

Figure 6-3 Input Data

(b) After inputting data, click “Run” button, we can get the results as shown in Figure 6-4(a) and 6-4(b).

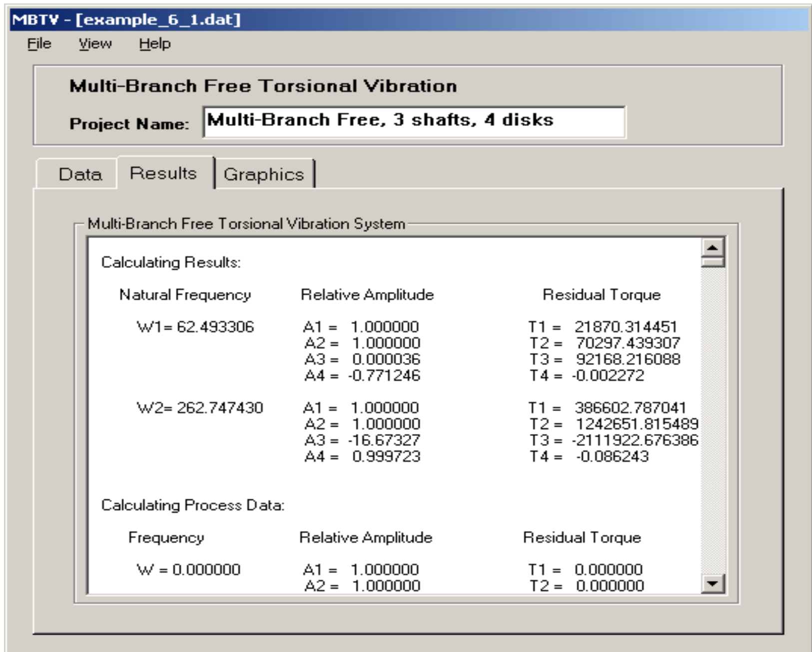

Figure 6-4(a) Results (ω_1 and ω_2)

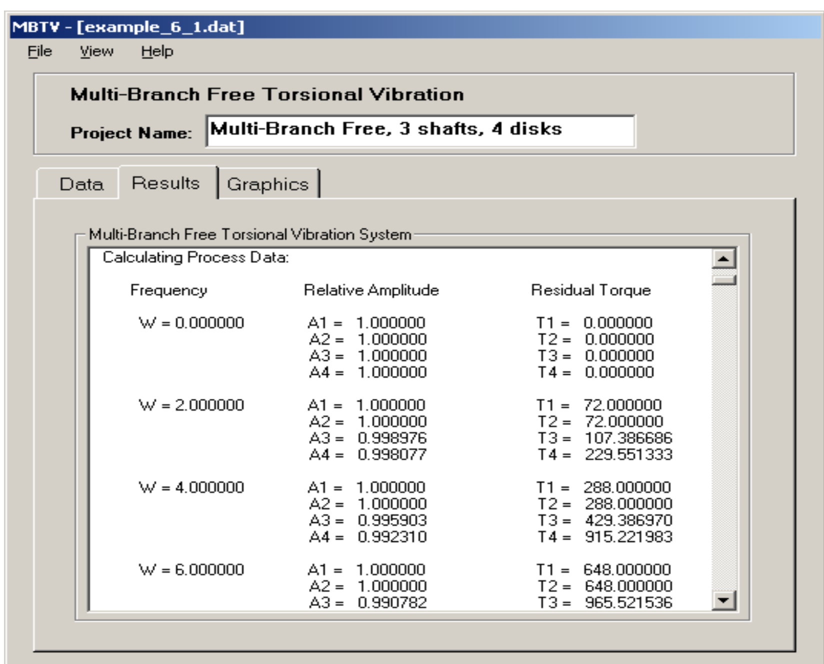

Figure 6-4(b) Results (ω increments)

The "Results" window will display the natural frequency of this vibration system in the range of frequencies specified in the "Calculating Process Data" window of Figure 6-3. The result of the relative amplitude of each disc is shown in Figure 6-4(a). Also, the calculating process in serial numbers of frequencies is demonstrated in Figure 6-4(b).

From Figure 6-4(b), we can know the variety of the residual torque of the end disc with different frequencies. With this, we can draw the curves of the residual torque vs frequencies. The frequencies are the natural frequencies of the vibration system, which make the residual torques of the end disc close to zero.

Specifying the data in the "Calculating Process Data" window, one will know the procedure and be able to evaluate more accurate results. For example, in Figure 6-5(a), we can enter in the "Beginning Frequency," "Ending Frequency," and "Increment of Frequency" values (62.493305, 62.493307, and 0.000001, respectively), as specified in the "Calculating Process Data" window. Then accurate results for this range can be obtained; this is shown in Figure 6-5(b).

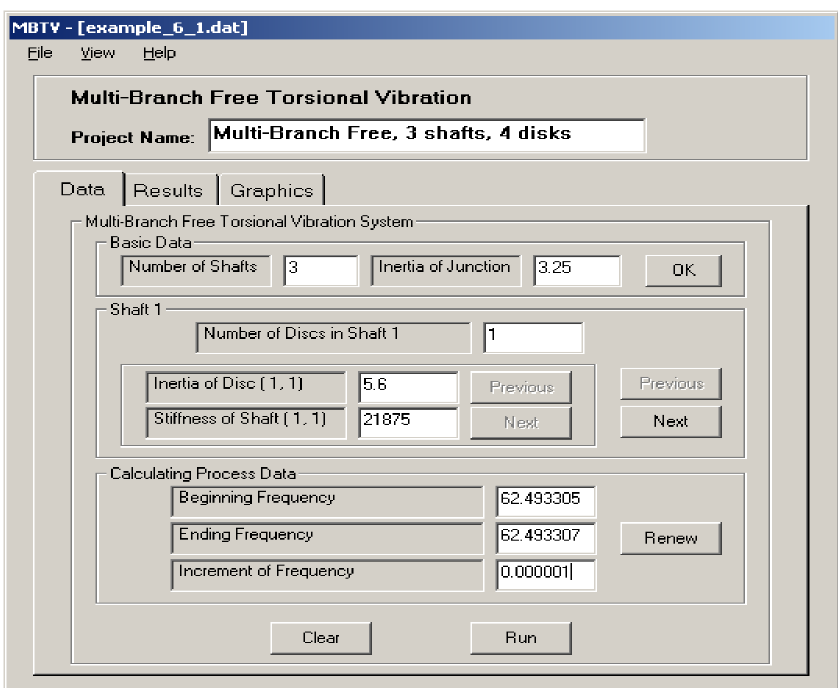

Figure 6-5(a) Specifying the Range for Natural Frequencies

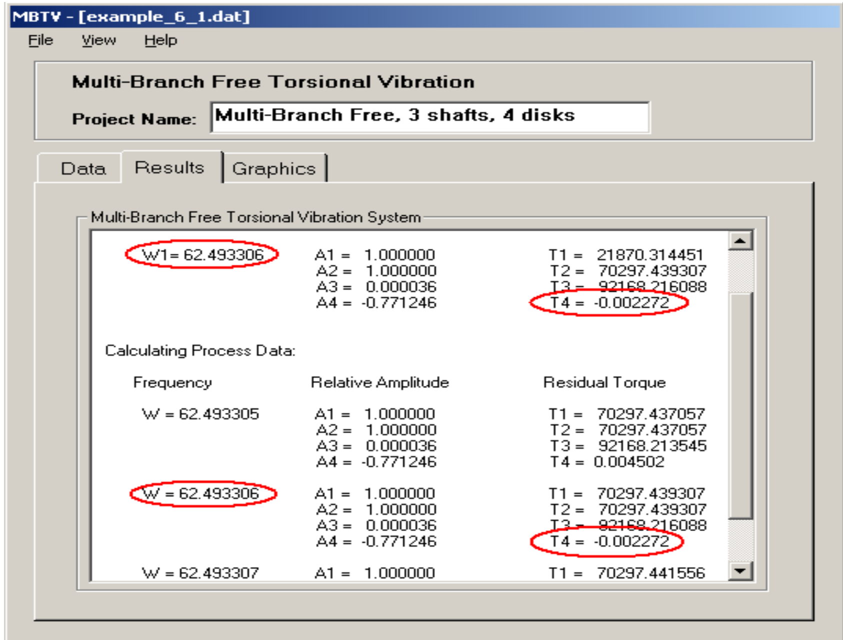

Figure 6-5(b) Calculating Process Results

The process results are only for users finding variations of the residual torque of the end disk with different frequencies. This provides additional information for users to understand procedure and calculations. If we do not specify any range in the "Calculating Process Data" window, we still can obtain the same correct results. However in that case, there will be no process data shown in "Results" window. The window will show the final results only. These windows are shown as Figure 6-5(c).

(c) Select "Graphics" button or "View / Graphics" command in Figure 6-5(b), MBTV will displays *T-W* Curve. These *T-W* relations are shown in Figure 6-6 and Figure 6-7.

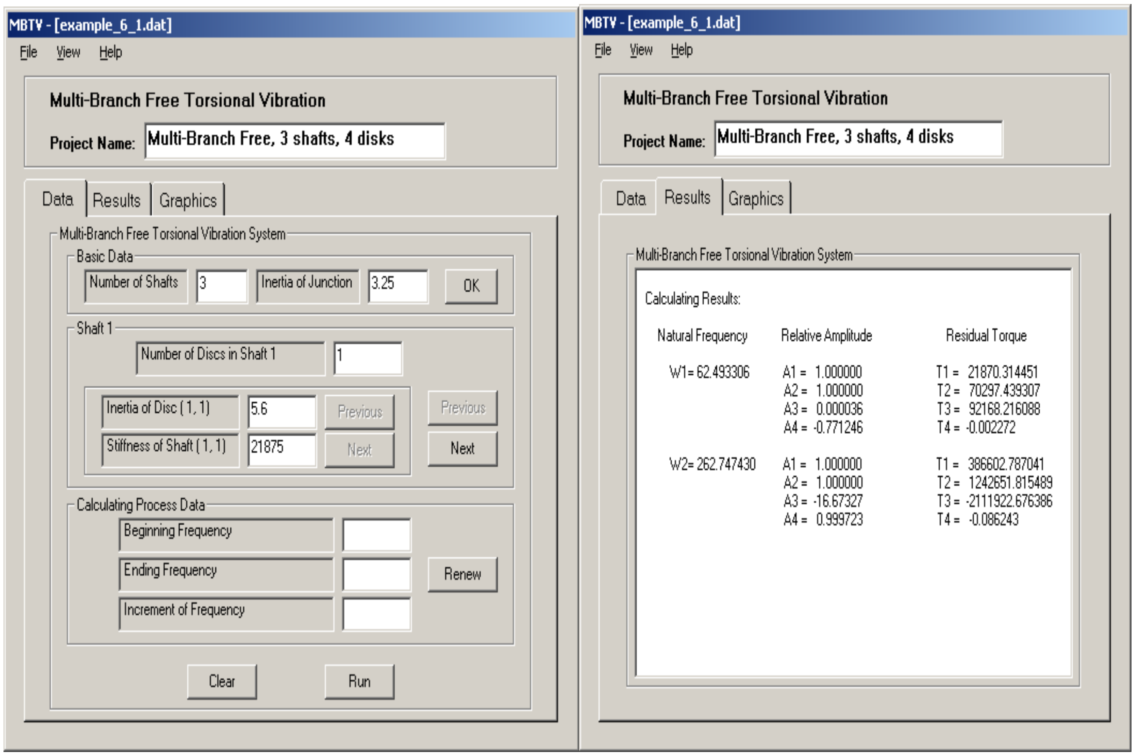

Figure 6-5(c) Result Without Showing Calculating Process

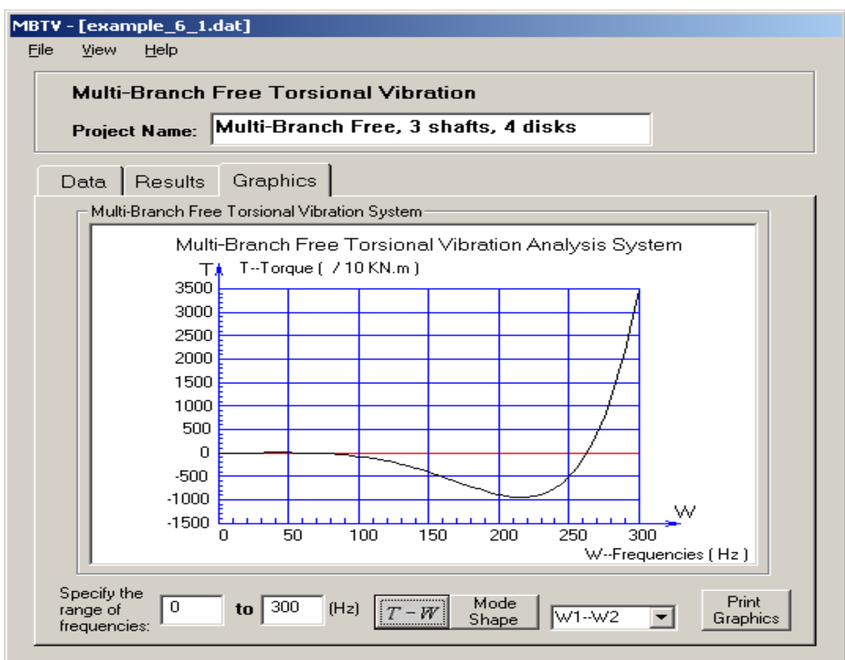

Figure 6-6 *T-W* Curve (0 ~ 300 Hz)

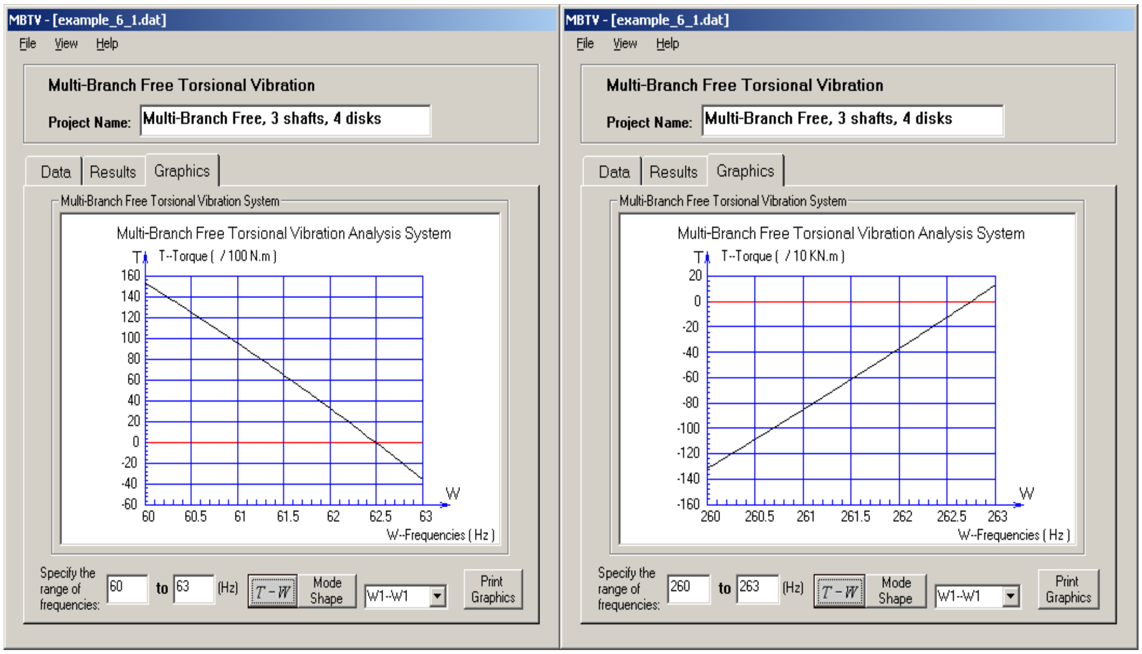

Figure 6-7 *T-W* Curves (60 ~ 63 Hz, 260 ~ 263 Hz)

(d) Click “Mode Shape” button, MBTV will display mode shape of the vibration system, shown in Figure 6-8 and Figure 6-9.

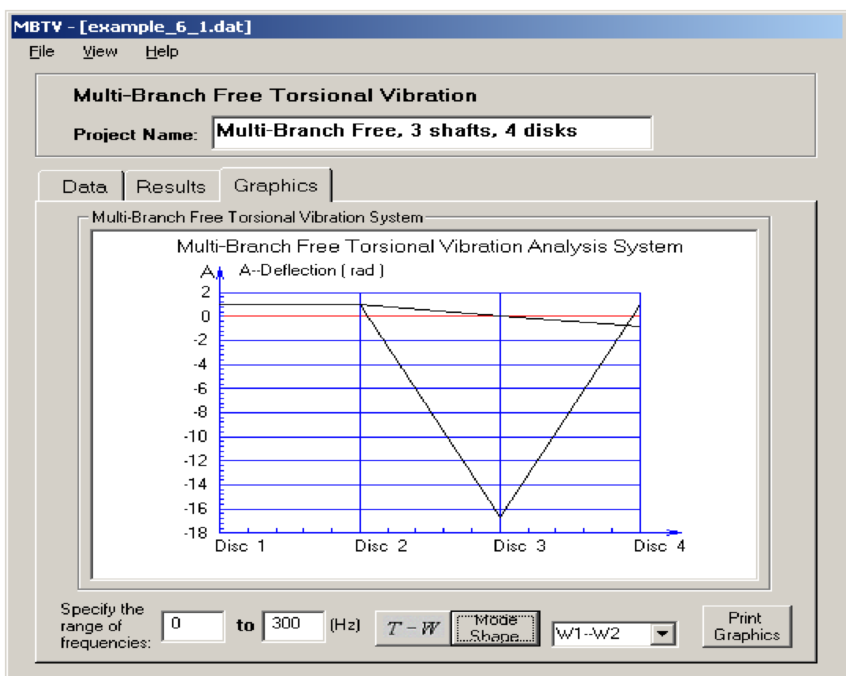

Figure 6-8 Mode Shapes

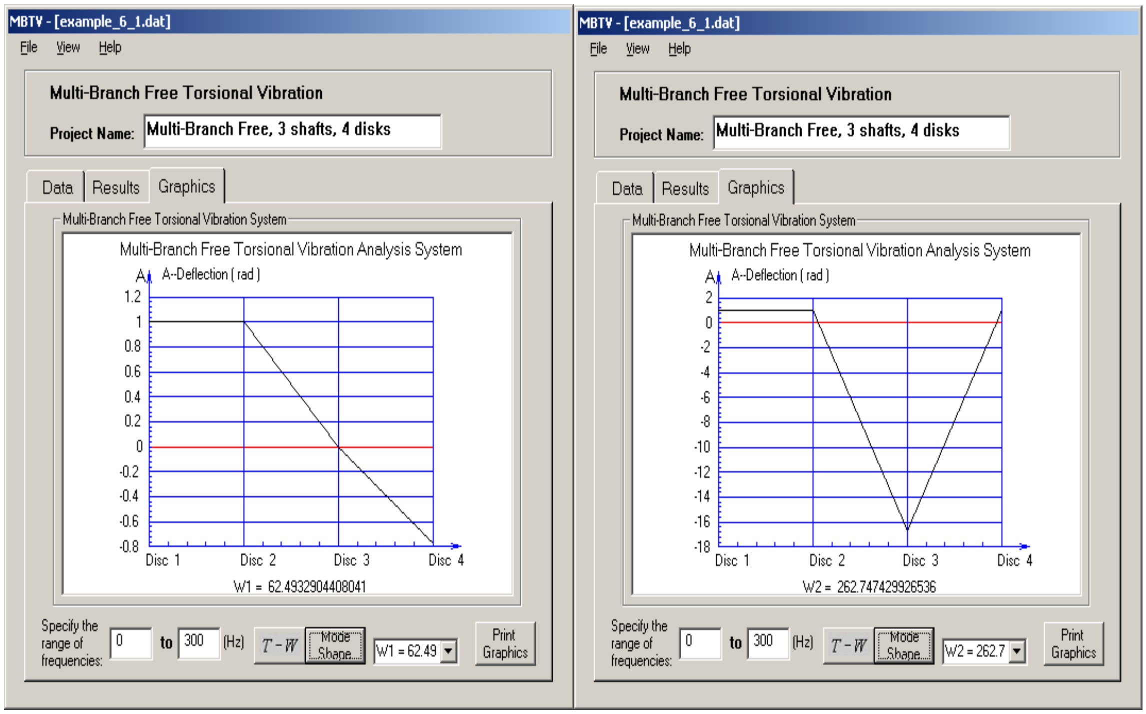

Figure 6-9 A Selected Mode Shape

4) Results Analysis

From Figure 6-4, the first natural frequency is ω_1=62.493306 (rad/s). At this frequency, the mode shape is as follows:

θ_1=1.000000

θ_2=1.000000

θ_3=0.000036

θ_4=-0.771246

The second natural frequency is ω_2=262.74743 (rad/s). At this frequency, the corresponding mode shape is as follows:

θ_1=1.000000

θ_2=1.000000

θ_3=-16.67327

θ_4=0.999723

The natural frequencies from Reference [42] are as follows.

$$\omega_1 = 62.5$$

$$\omega_2 = 263$$

The relative differences between MBTV and reference are as follows.

$$\omega_1 \quad : \quad \frac{|62.493306 - 62.5|}{62.493306} = 0.011\%$$

$$\omega_2 \quad : \quad \frac{|262.74743 - 263|}{262.74743} = 0.009\%$$

The relative differences are very small. The results from different calculations are very close.

6.2 A Steel Rolling Machine

The following sketch shown in Figure 6-10 is a main transmission system of a steel rolling machine [41]. We shall test and analyze the vibration characteristic of this main transmission system. The testing scheme is shown in Figure 6-11.

The results calculated by MBTV program will be compared with the tested results.

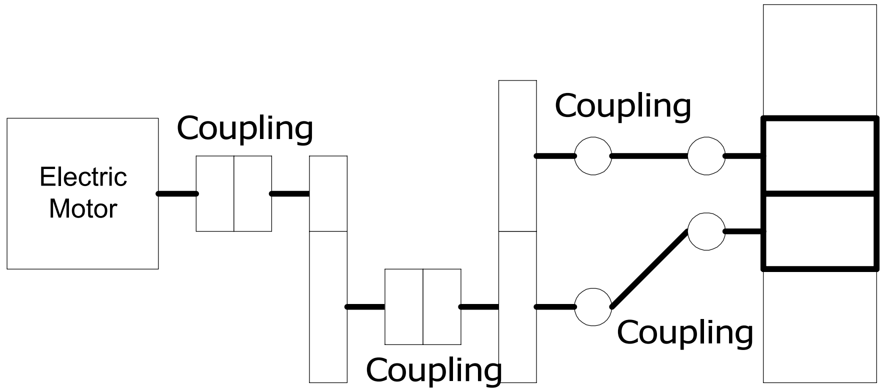

Figure 6-10 Transmission System of a Steel Rolling Machine

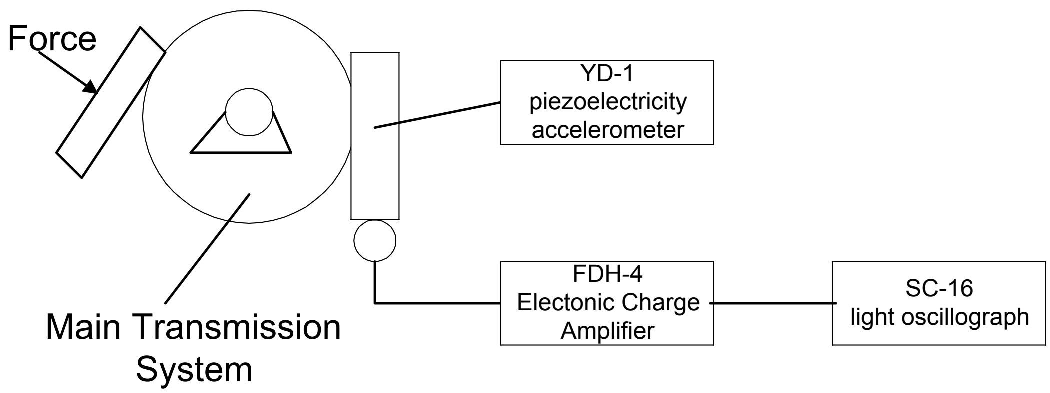

Figure 6-11 Testing Scheme

■ **Solution Steps**

1) Determine the number of branches and the values of inertia and stiffness in each branch. Set one branch as reference and input the speed ratio for each branch with regard to the reference branch.

In this case, take the shaft of the electric motor as a reference branch and the equivalent system can be described as a multi-branch, single junction, free vibrations without damping. This equivalent system is shown in Figure 6-12.

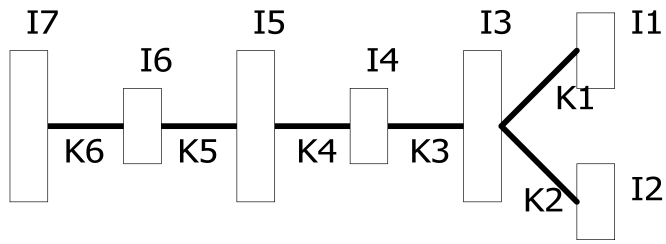

Figure 6-12 Equivalent System

2) Calculate the equivalent values of inertia I_i and stiffness K_i relevant to the reference branch.

$$K_1 = K_2 = 44192 N \cdot m / rad$$

$$K_3 = 148740 N \cdot m / rad$$

$$K_4 = 1.1632\mathrm{e}+6 N \cdot m / rad$$

$$K_5 = 2.62\mathrm{e}+6 N \cdot m / rad$$

$$K_6 = 7.59\mathrm{e}+6 N \cdot m / rad$$

$$I_1 = I_2 = 0.07245 m^2 Kg$$

$$I_3 = 0.04196 m^2 Kg$$

$$I_4 = 0.20620 m^2 Kg$$

$$I_5 = 0.3811 m^2 Kg$$

$$I_6 = 1.1721 m^2 Kg$$

$$I_7 = 54.768 m^2 Kg$$

3) Run Program MBTV to calculate the values of the residual torques with different frequencies, and obtain the results including natural frequencies and mode shape of this vibration system.

(a) Input the structure data and characteristic parameters, shown in Figure 6-13.

(a) After inputting data, click “Run” button, we can get the results as shown in Figure 6-14.

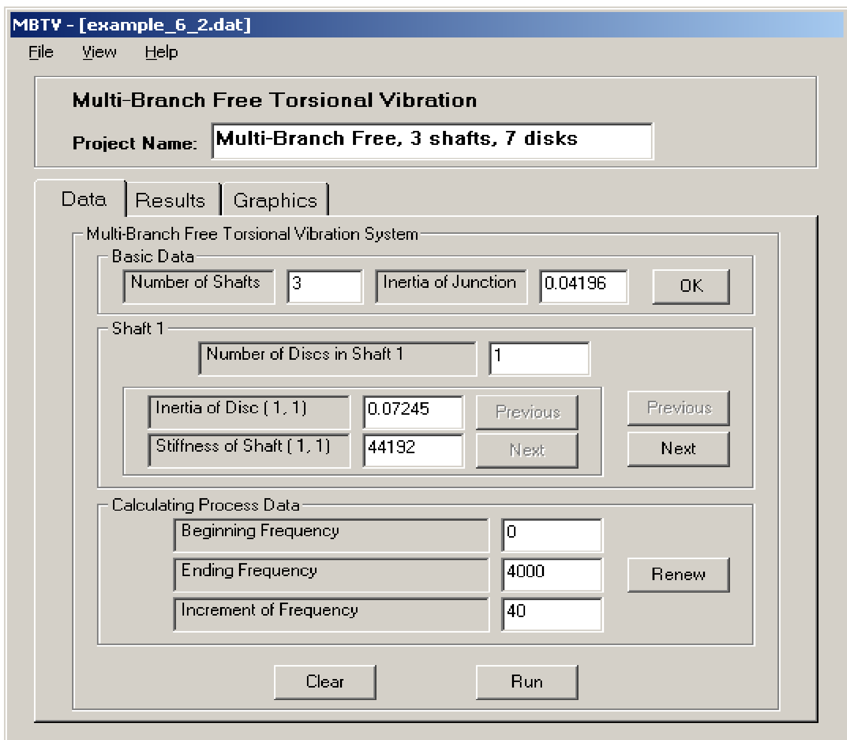

Figure 6-13 Input Data

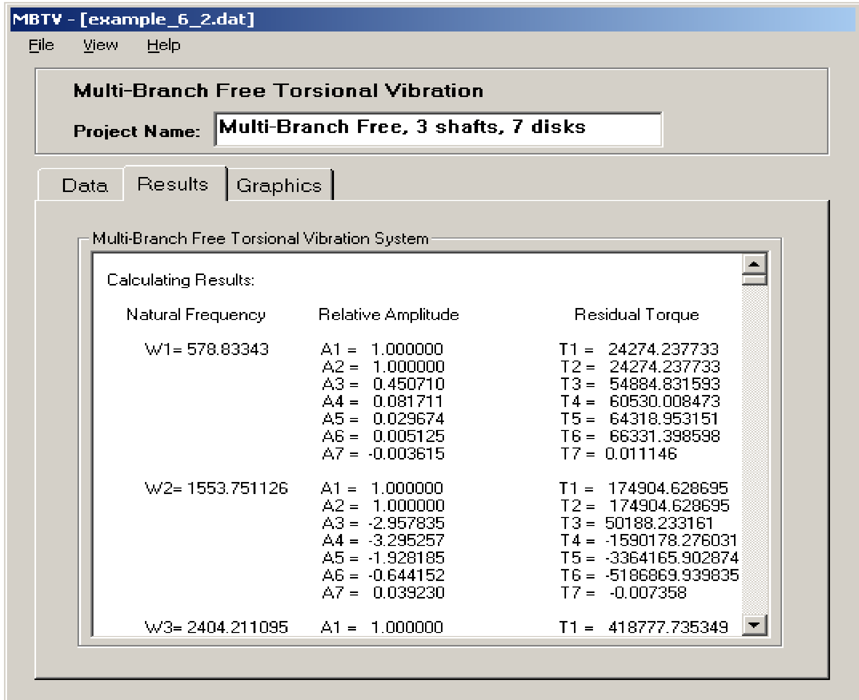

Figure 6-14(a) Results (ω_1 ~ ω_3)

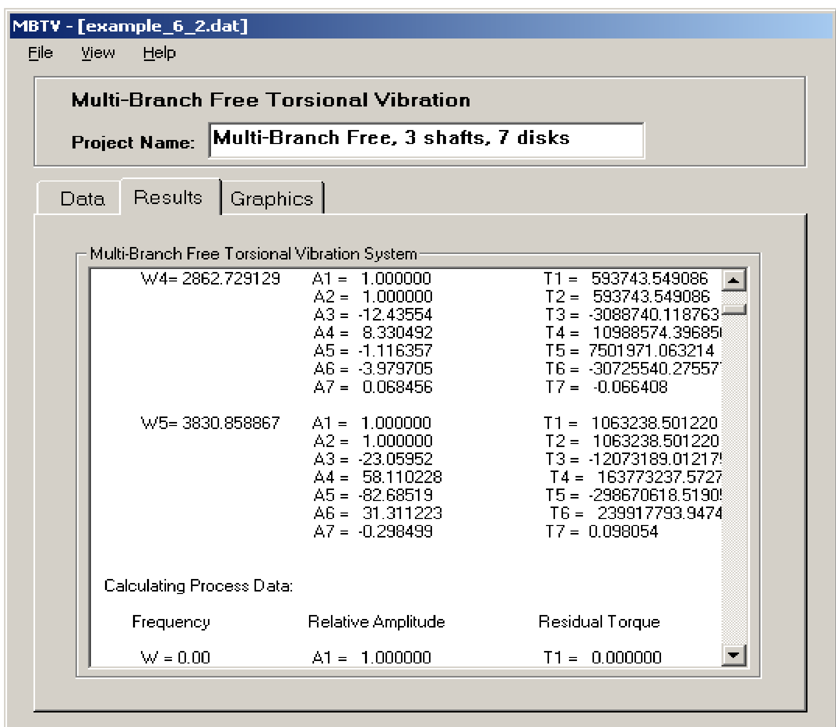

Figure 6-14(b) Results (ω_4 and ω_5)

In this engineering case, there are five natural frequencies. They are:

$$\begin{cases} \omega_1 = 578.83343 \\ \omega_2 = 1553.7511 \\ \omega_3 = 2404.2111 \\ \omega_4 = 2862.7291 \\ \omega_5 = 3830.8589 \end{cases}$$

If we want to review an accurate natural frequency, we can specify a small range of frequencies and define a small increment of frequency from Figure 6-13. These can be completed from Figure 6-13 as previously described, and the results are demonstrated in Figure 6-15. This shows how the first natural frequency $\omega_1 = 578.83343$ is obtained.

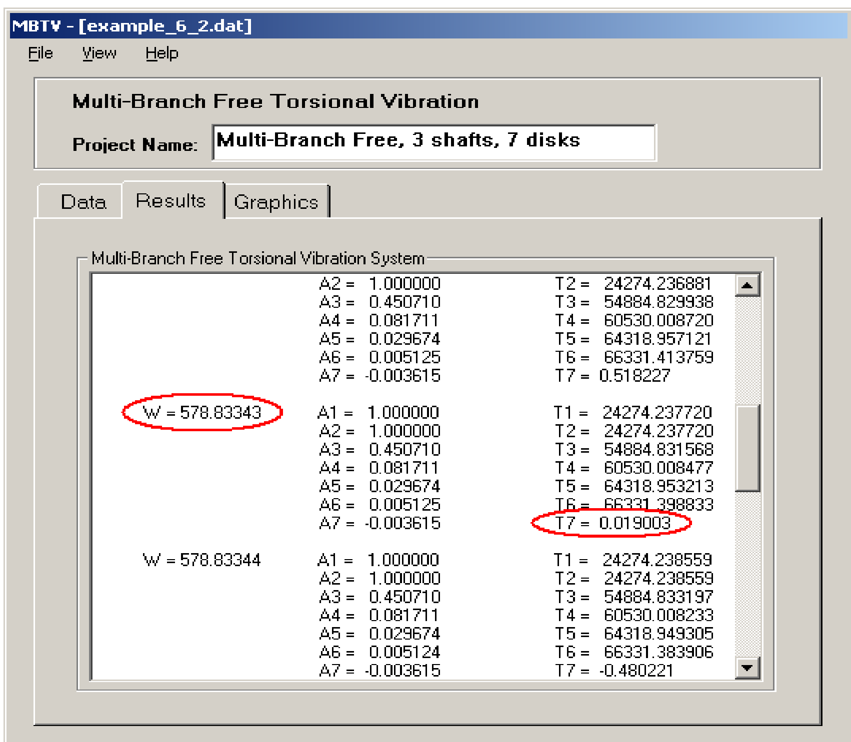

Figure 6-15 Calculating process Results

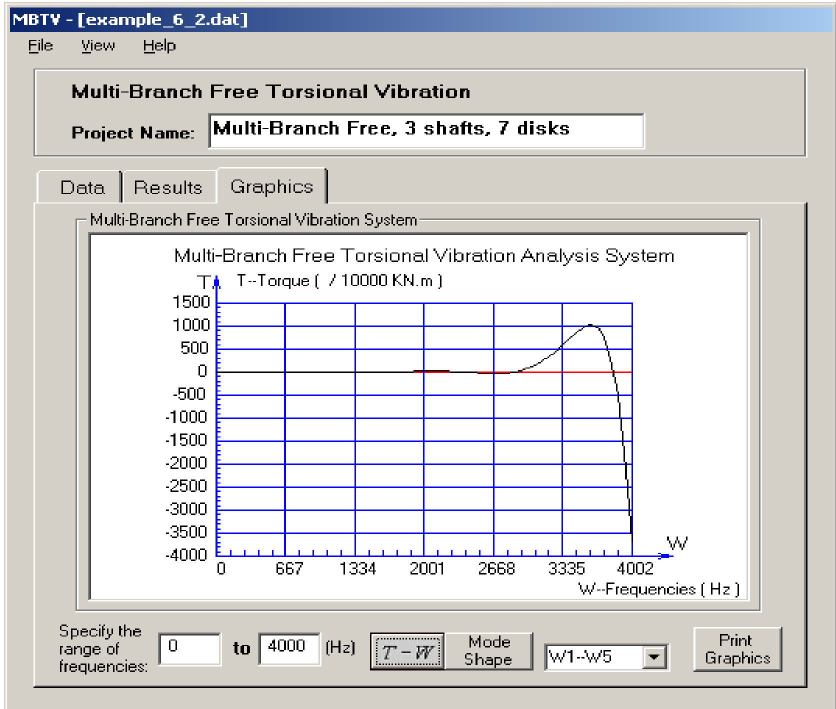

Figure 6-16 $T-W$ Curve (0 ~ 4000 Hz)

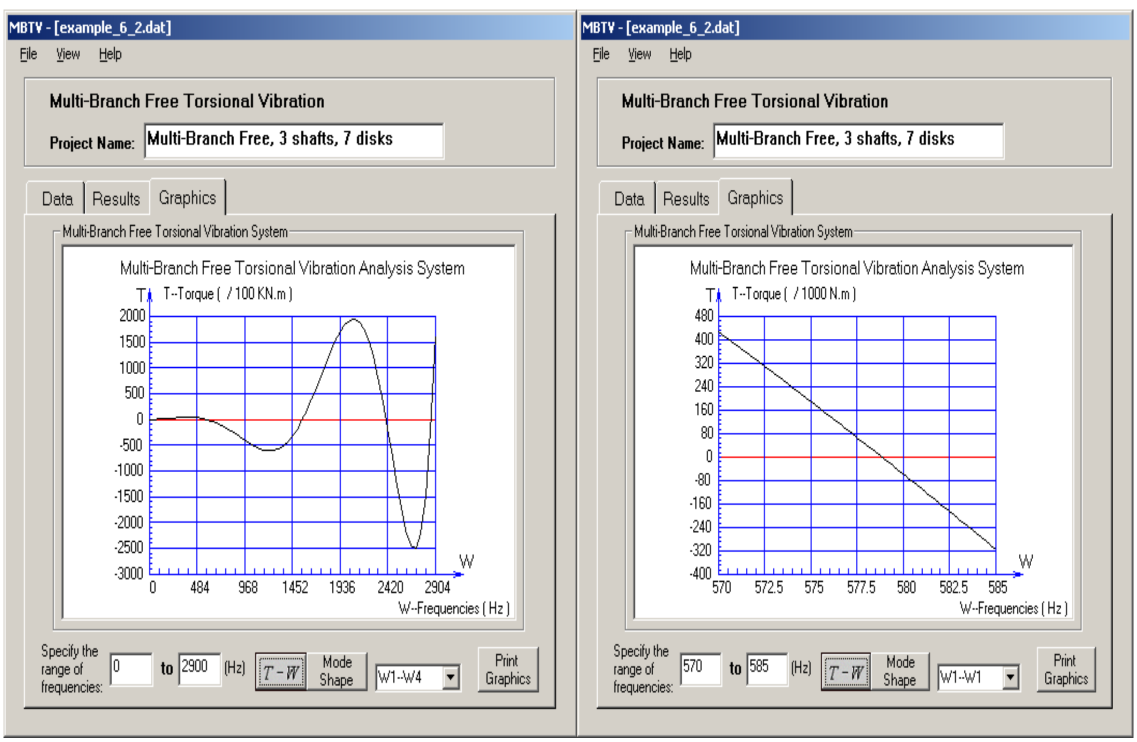

Figure 6-17 $T-W$ Curve (0 ~ 2900 Hz, 570 ~ 585 Hz)

(c) Select “Graphics” button or “View/Graphics” command in Figure 6-15, MBTV will display *T-W* Curves. These *T-W* relations are shown in Figure 6-16 and Figure 6-17.

(d) Click “Mode Shape” button, MBTV will display mode shapes of the vibration system as shown in Figure 6-18(a). Also, the software allows displaying any particular mode shape by selecting the corresponding natural frequency from the window next to “Mode Shape” button. This function is shown in Figure 6-18 (b).

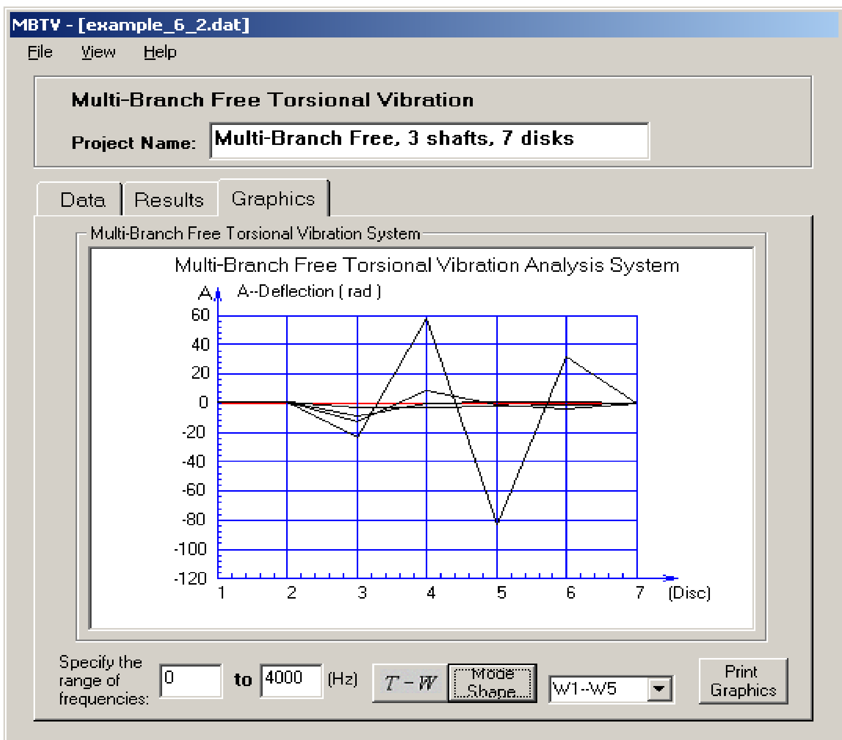

Figure 6-18(a) Mode Shapes

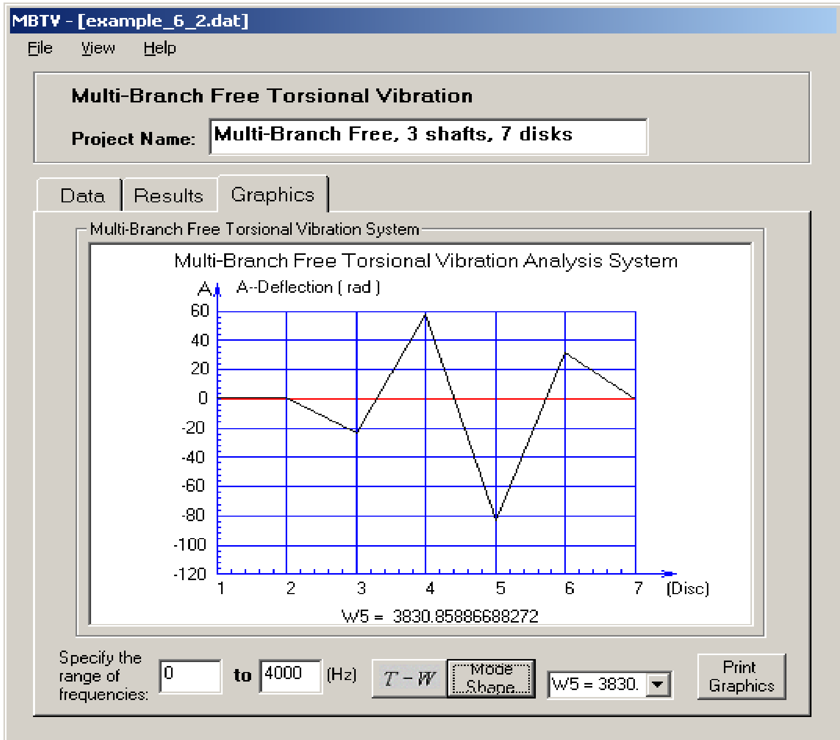

Figure 6-18(b) Display of a Selected Mode Shape

4) Results Analysis

From the Figure 6-14, we can get the first natural frequency. ω_1 =578.83343 (rad/s). At this frequency, the corresponding mode shape is as follows:

θ_1=1.000000

θ_2=1.000000

θ_3=0.450710

θ_4= 0.081711

θ_5= 0.029674

θ_6= 0.005125

θ_7=-0.003615

From the Figure 6-14, we can also get the second natural frequency. ω_2=1553.751126 (rad/s). At this frequency, the corresponding mode shape is as follows.

θ_1= 1.000000

θ_2=1.000000

θ_3=-2.957835

θ_4= -3.295257

θ_5= -1.928185

θ_6= -0.644152

θ_7= 0.039230

To compare with the test results, the natural frequencies of this vibration system are $\omega_1 = 579.839946$, $\omega_2 = 1554.05565$ (Reference [41]). The relative differences are as follows:

The First Frequency:

$$\frac{|578.83343 - 579.83994|}{578.83343} = 0.18\%$$

The Second Frequency:

$$\frac{|1553.751126 - 1554.05565|}{1553.751127} = 0.02\%$$

The relative differences are very small. The results from different calculations are very close.

6.3 A Free Torsional Vibration System with Damping

Consider an in-line torsional vibration system with damping [35] as shown in Figure 6–19. Determine the torque - frequency curve and establish the amplitude and residual torque for each station when the specified frequency is 31.6Hz.

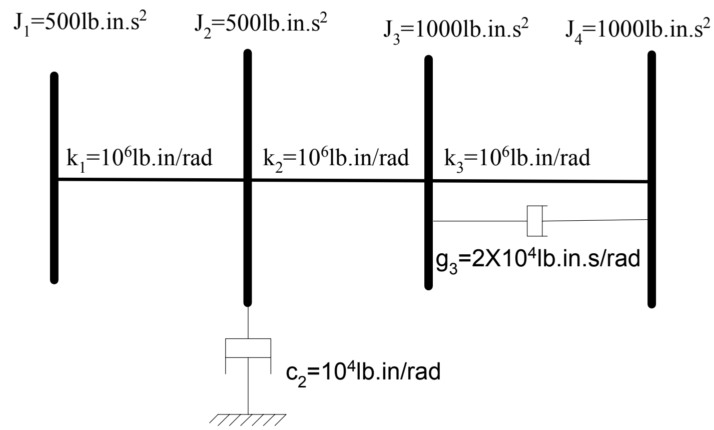

Figure 6-19 Free Torsional Vibration with Damping

- Calculations and Solution

Run Program MBTV to calculate the values of the residual torques with different frequencies.

(a) Input the structure data and characteristic parameters, shown in Figure 6-20(a). Comparing Figure 6-20(a) with Figure 6-3, 6-13, we can find that input items and commands vary with analysis types.

(b) Click “Run” button, we can get the results as shown in Figure 6-20(b).

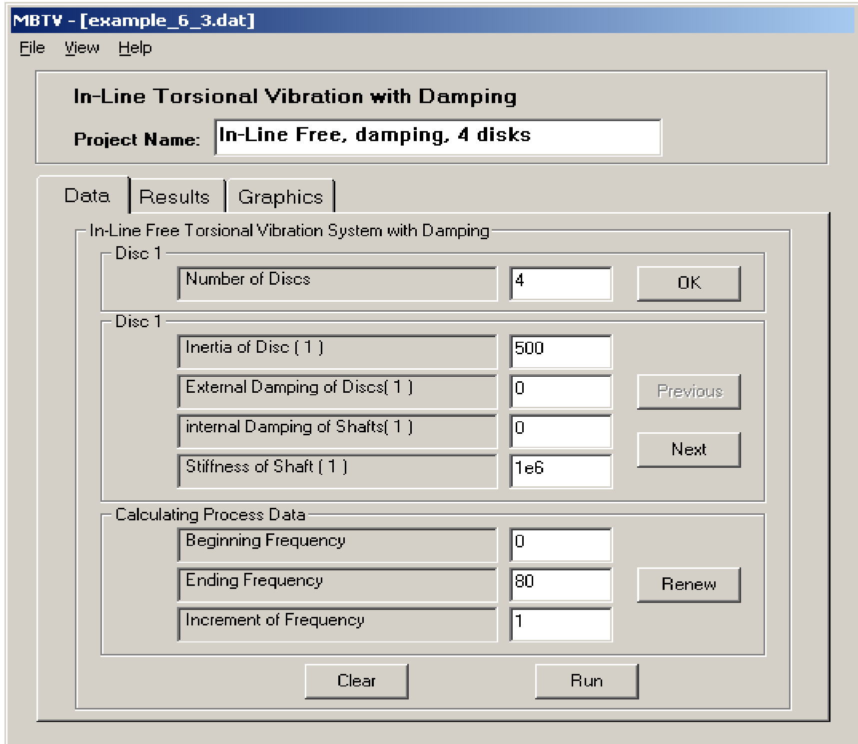

Figure 6-20(a) Input Data

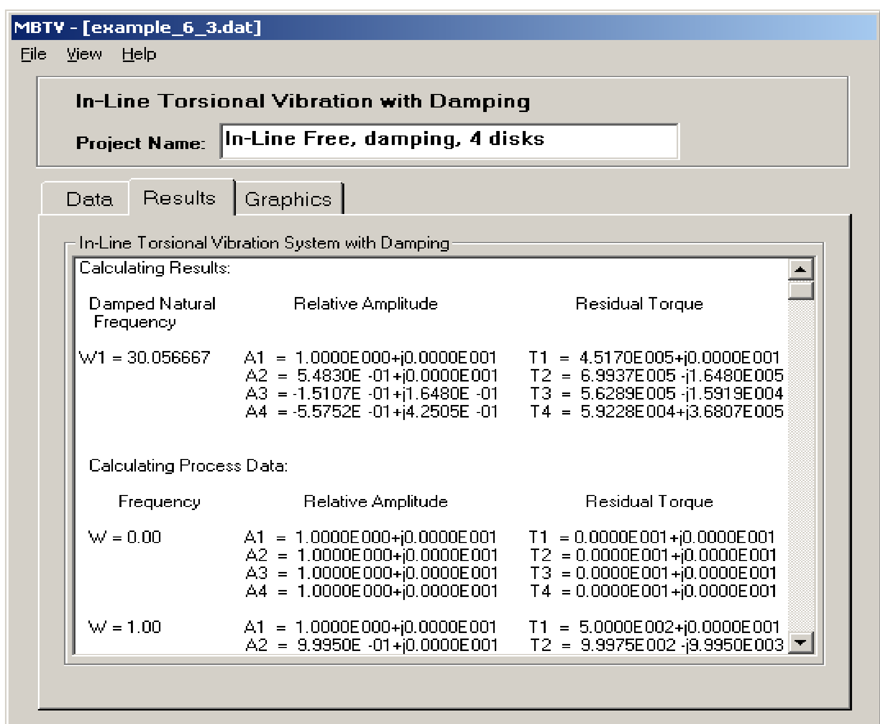

Figure 6-20(b) Results

In the above figure, we can find that there is only one damped natural frequency. It is 30.056667 rad/s.

We can also calculate the amplitude and residual torque of each station under the specified frequency. We will calculate the amplitude under $\omega^2 = 1000, \ \ i.e,$ $\omega \cong 31.6228(rad/s)$. The input data is now shown in Figure 6-21. Select the Run button, we can obtain the results as shown in Figure 6-22.

(c) Select "Graphics" button or "View / Graphics" command in Figure 6-22, MBTV will display *T-W* Curve as shown in Figure 6-23.

(d) Click "*T-W* ^{2}" button, MBTV will display *T-W* ^{2} Curve as shown in Figure 6-24. By specifying different ranges of frequency, we can obtain clearer *T-W* ^{2} Curves as shown in Figure 6-25. This feature makes analysis more convenient when detailed evaluations are needed.

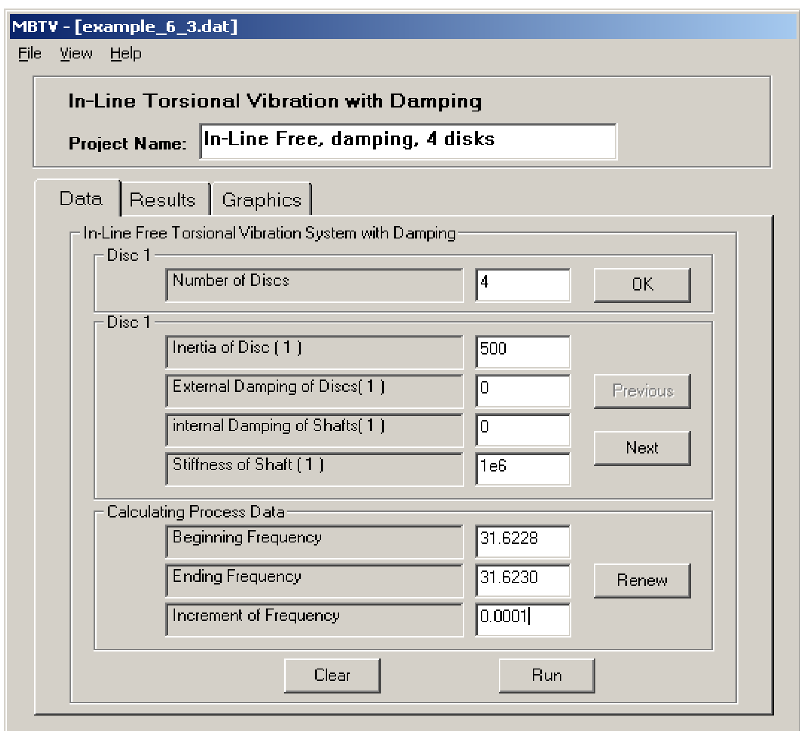

Figure 6-21 Specify a Range

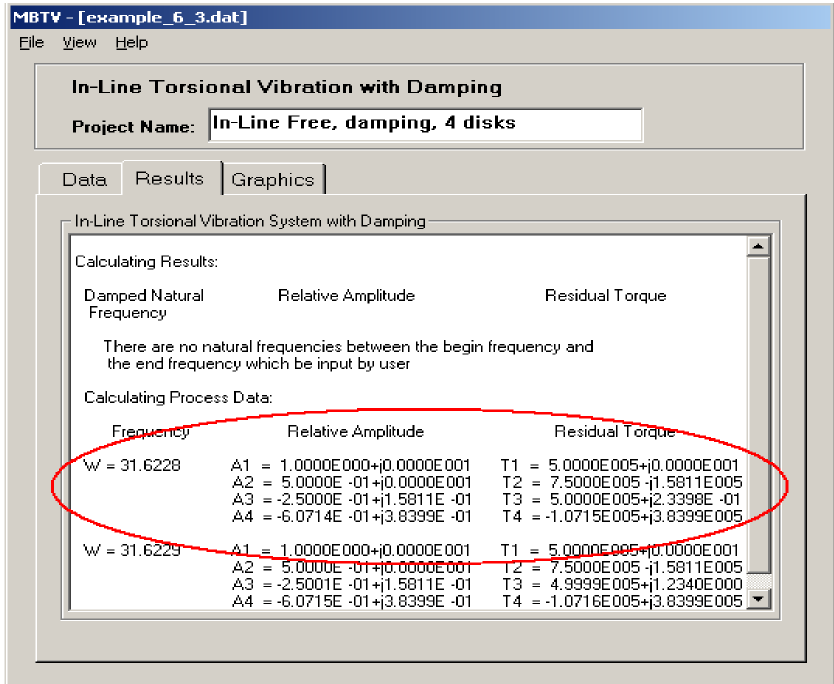

Figure 6-22 Results for Specified Frequencies

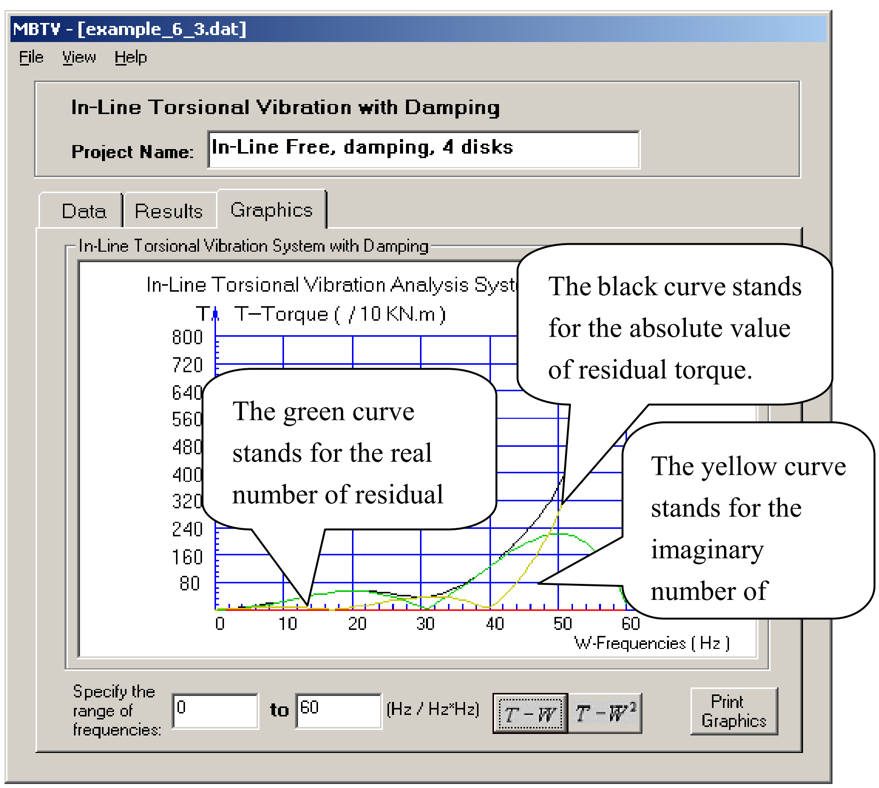

Figure 6-23 $T-W$ Curve

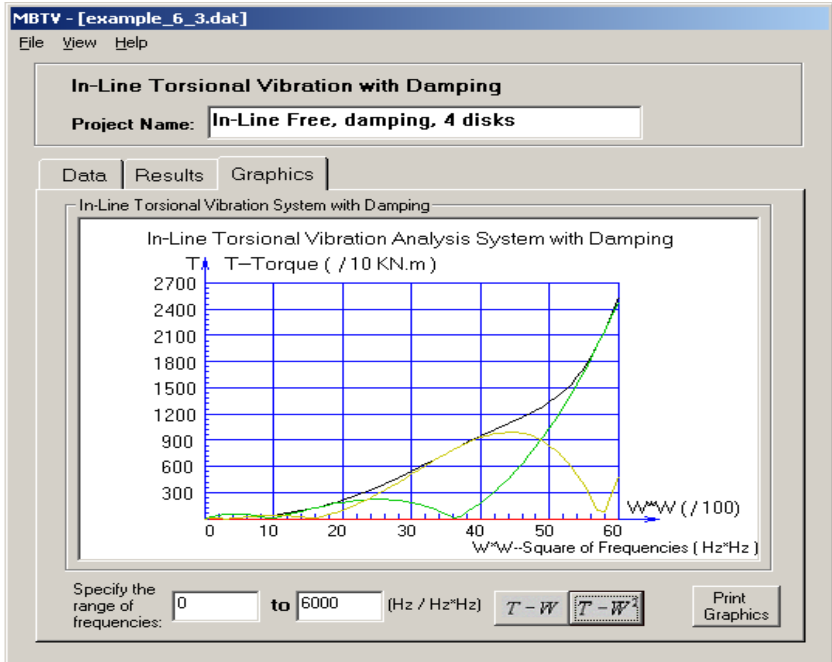

Figure 6-24 $T - W^2$ Curve (0 ~ 6000 Hz)

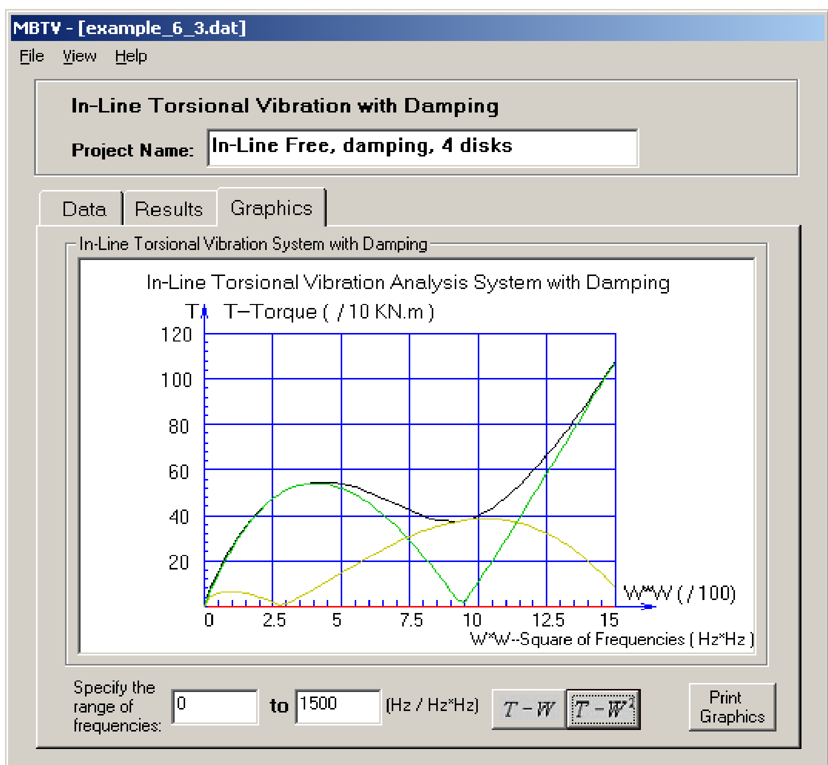

Figure 6-25 $T - W^2$ Curve (0 ~ 1500 Hz)

- Results Analysis

1) The results by MBTV at specified frequency,

$\omega^2 = 1000,\ \ i.e,\ \ \omega \cong 31.6228(rad/s)$, shown as Figure 6-22.

$$\begin{cases} \theta_1 = 1.0000E000 + j0.0000E001 \\ \theta_2 = 5.0000E-01 + j0.0000E001 \\ \theta_3 = -2.5000E-01 + j1.5811E-01 \\ \theta_4 = -6.0714E-01 + j3.8399E-01 \end{cases}$$

The magnitude of θ_4 is:

$$\left|\theta_4\right| = \sqrt{(-0.60714)^2 + (0.38399)^2} = 0.718378(rad)$$

2) The results from Reference [35]:

θ_1 =1.0+j0.0

θ_2 =0.50+j0.0

θ_3 = -0.25+j0.158

θ_4 =-0.607+j0.384

The magnitude of θ_4 is

$$\left|\theta_4\right| = \sqrt{(-0.607)^2 + (0.384)^2} = 0.718265(rad)$$

3) From the above results, the maximum relative difference is

$$\theta_4: \quad \frac{\left|0.718378 - 0.718224\right|}{0.718378} = 0.002\%$$

The relative difference is very small. The results from different calculations are very close.

- Discussion

In a free torsional vibration system, the end torque or residual torque is a real number, which means that the torque is in phase with motion at every disc. In the damped torsional vibration system, the residual torque is a complex quantity, containing components in phase with and in quadrature to the motion at the other discs. In the free vibration case, there are certain frequencies for which the residual torque becomes zero, which means that the system can have a steady-state vibration without any external excitation. With the presence of damping, this obviously can no longer be the case. The residual torque never becomes zero, but for the certain values of frequencies, it becomes a minimum. We define these frequencies as the “damped natural frequencies”. For small damping system, they differ but slightly from the free vibration, true, natural frequencies.

The magnitude of damping will affect the magnitude of the “damped natural frequencies”. For example, in this system, if we change g_3 from 2×10^4 $lb\cdot in\cdot s/rad$ to 2×10^2 $lb\cdot in\cdot s/rad$. There will be two “damped natural frequencies”, which becomes 2 from 1. The first damped natural frequency will become 29.083333 rad/s from 30.056667 rad/s. (Refer to Figure 6-26 and Figure 6-27). The difference can be found by comparing Figure 6-26 with Figure 6-21 and Figure 6-27 with Figure 24.

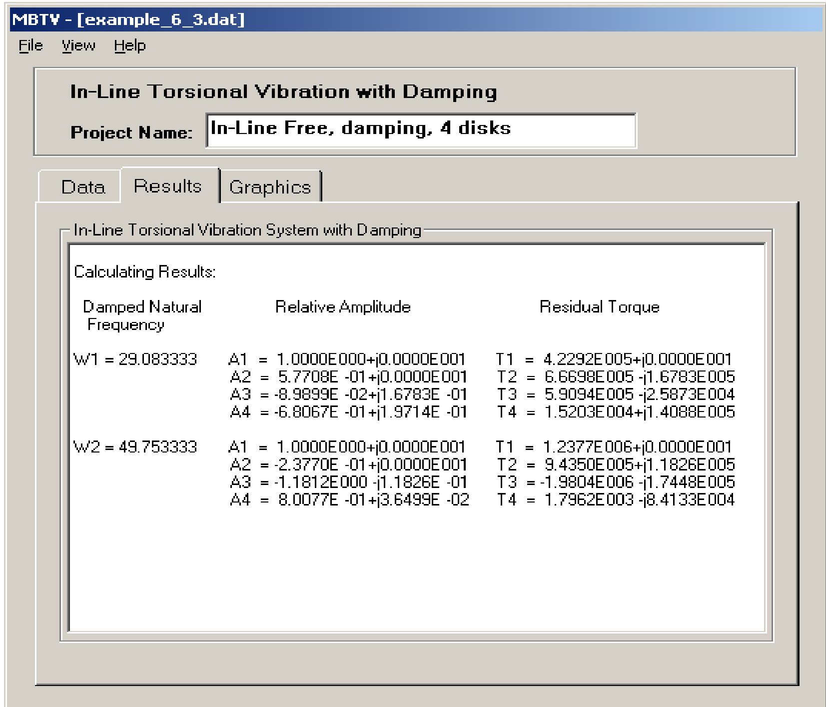

Figure 6-26 Results After Changing g_3 to 2×10^2

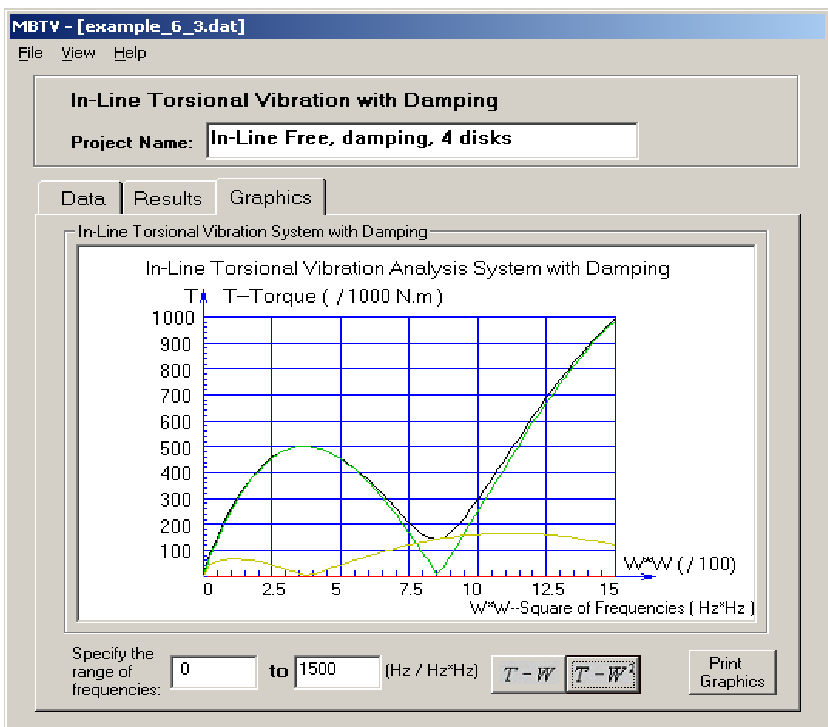

Figure 6-27 T-W^2 Curve After Changing g_3 to 2×10^2

If we remove these two dampers in this system, i.e., let $c_2 = 0$ and $g_3 = 0$, the system will become a free vibration system without damping. In this case, it will be a free vibration system. There will be three natural frequencies. The results are shown in Figure 6-28, 6-29 and 6-30.

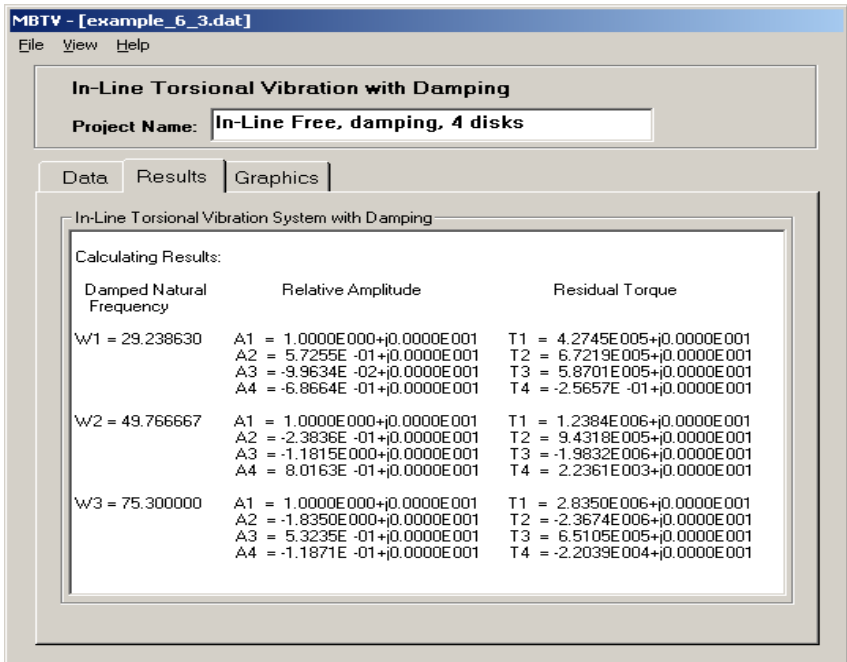

Figure 6-28 Results After Removed Dampers

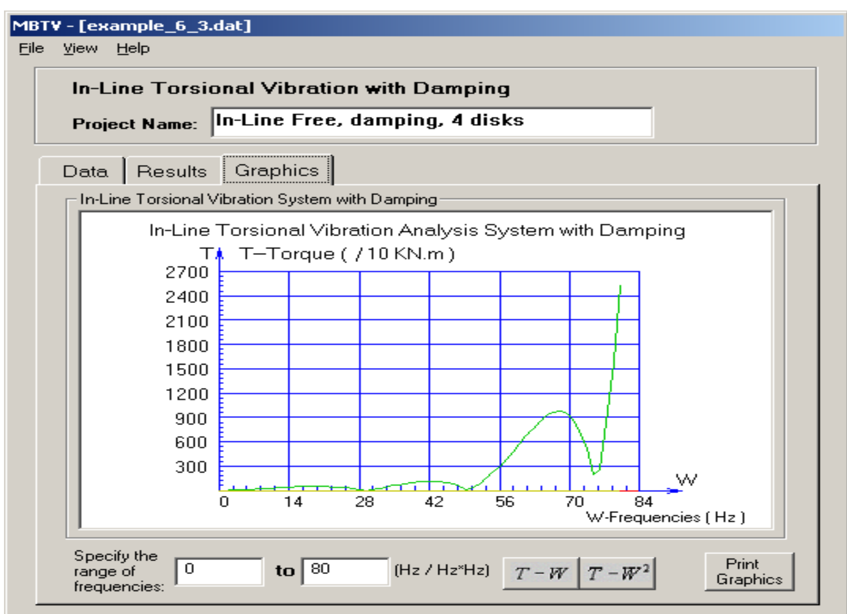

Figure 6-29 *T-W* Curve After Removed Dampers

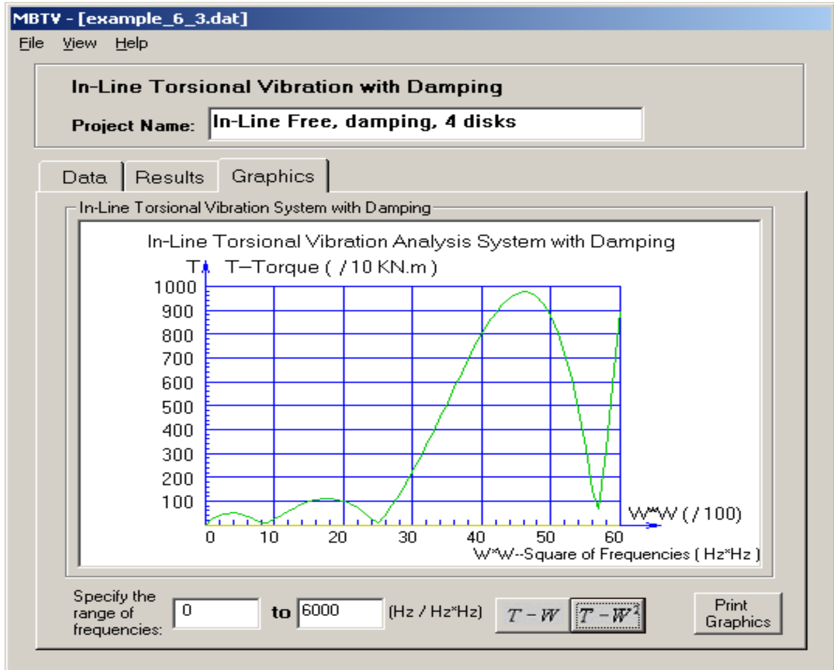

Figure 6-30 $T\text{-}W^2$ Curve After Removed Dampers

6.4 A Forced Torsional Vibration System with Damping

The torsional vibration system of Figure. 6-31 is excited by a harmonic torque T_4 at a point to the right of disc 4. There is no any torque on other three discs, i.e. $T_1 = 0$, $T_2 = 0$, $T_3 = 0$, $T_4 = 2000\sin(31.6t)$, $\varphi_4 = 0$. Determine the amplitude and residual torque for each station [35]. The parameters are shown in Figure 6-31.

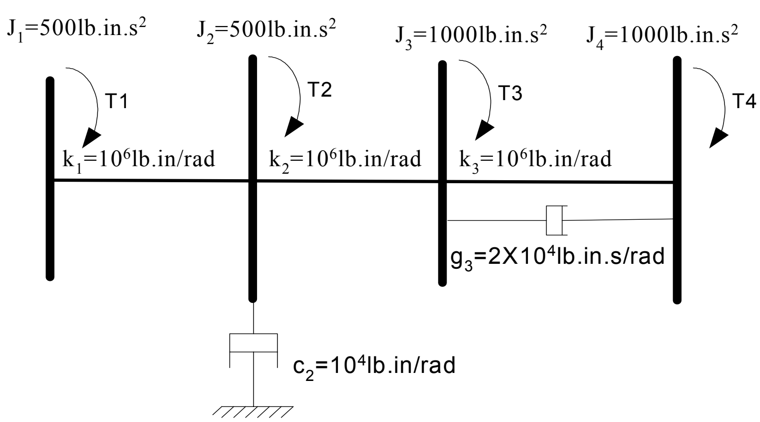

Figure 6-31 An In-line Forced Torsional Vibration System with Damping

■ Calculations and Solution

Run Program MBTV to calculate the values of the residual torques with different frequencies.

(a) Input the structure data and characteristic parameters, shown in Figure 6-32.

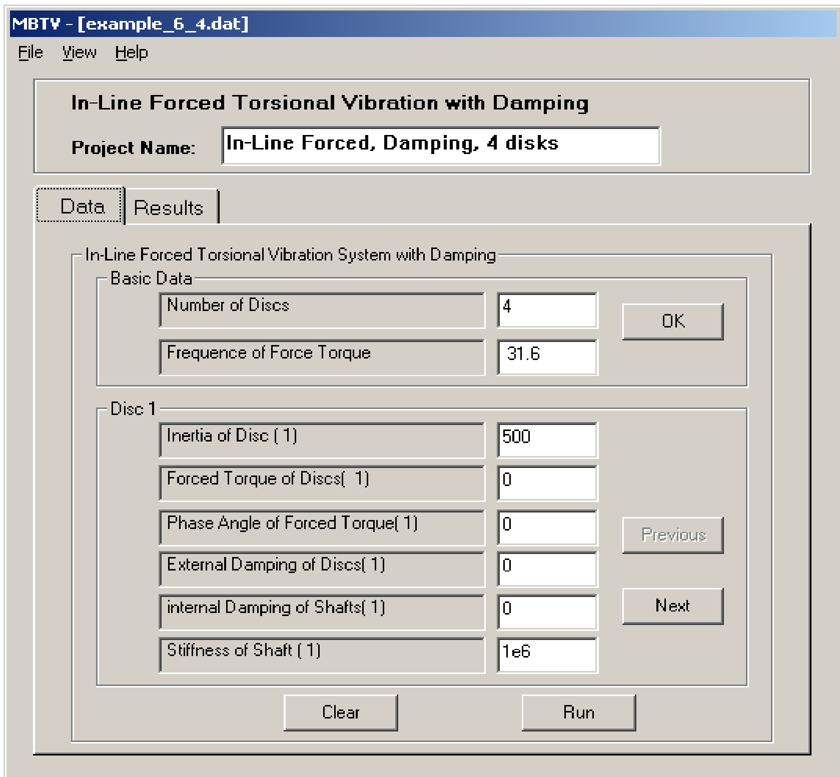

Fig. 6-32 Input Data

(b) After inputting data, click “Run” button. We can get the results as shown in Figure 6-33.

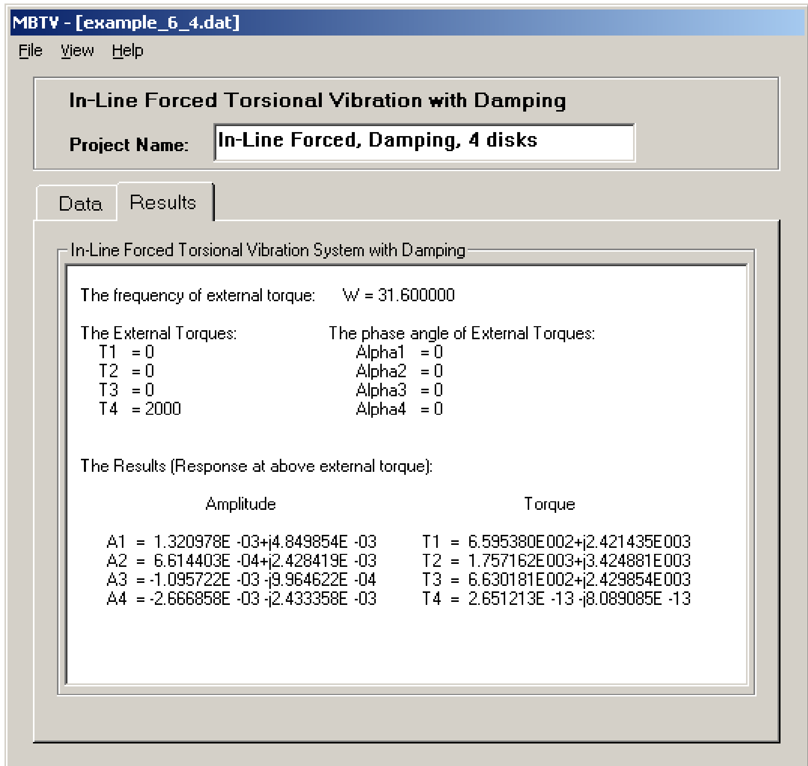

Figure 6-33 Results

In Figure 6-33, symbol “alpha” represents the preliminary angles of the external torques.

From above, we can find the torque of end disc on shaft close to zero. It accords with the theory, which has been described in Section 4.4.

■ Results Analysis

1) The results by MBTV are summarized as follows, according to Figure 6-33

ω=31.6227 (ω^2=1000)

$$\theta_2 = 6.614403\text{E}-04 + \text{j}2.428419\text{E}-03$$

that is: $\theta_2 = \sqrt{(0.0006614403)^2 + (0.002428419)^2} = 0.002516887(rad)$

2) The results from Reference [35].

ω =31.6 (ω^2 =1000)

θ_2 = 0.00254

3) The relative difference

$$\theta_2 \qquad \frac{|0.00254-0.002516687|}{0.00254}=0.92\%$$

The relative difference is very small. The results from different calculations are very close.

■ Discussion

In Figure 6-33, the A1 to A4 are respectively the amplitudes of Disc1 to Disc4 under exciting by the external torque $T_4 = 2000\sin(31.6t+\varphi_4),\ \ \varphi_4 = 0$. T1 to T3 are respectively the torques of shaft1 to shift3 under exciting by the external torque $T_4 = 2000\sin 31.6t$. Shaft1 means the shaft segment between Disc1 to Disc2. We can calculate the stress and strain of the shaft through the torques. By these results, we can find out the designed system whether it meets engineering demands or not.

6.5 A Forced Torsional Vibration System without Damping

A four rotor system is represented schematically in Figure 6-34 (reference [43]). the physical parameters of the system are shown on it. An external stimulant torque $T=T_0 \sin \omega t$ (T_0=10000 N.m, ω=5 rad/s) acts on the second rotor. Other external torques are equal to zero, i.e., $T_1 = T_3 = T_4 = 0,\ \ T_2 = 10000\sin(5t)\ \ (Nm)$. Determine the amplitude and residual torque of vibration of various rotors.

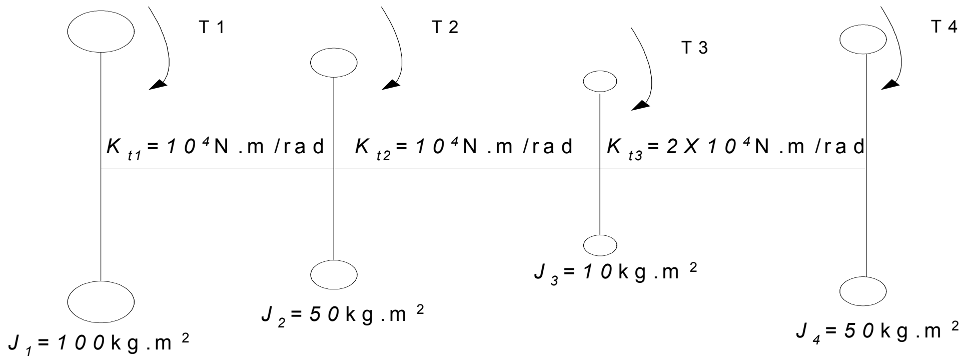

Figure 6-34 An In-line Forced Torsional Vibration system without Damping

■ Calculations and Solution

Run Program MBTV to calculate the values of the residual torques with different frequencies.

(a) Input the structure data and characteristic parameters, shown in Figure 6-35.

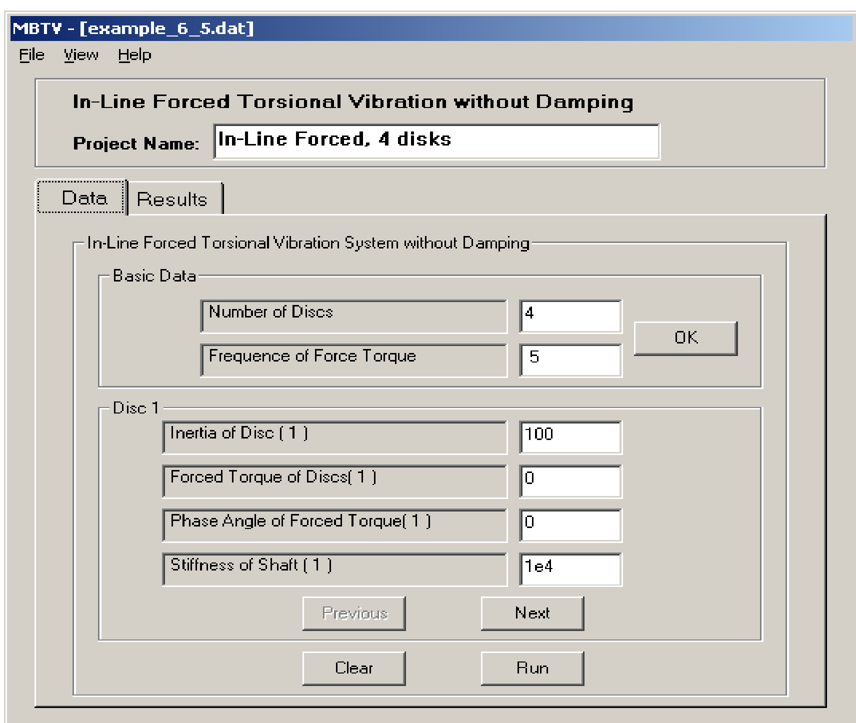

Figure 6-35 Input data

(b) After inputting data, click “Run” button, we get the results, shown in Figure 6-36.

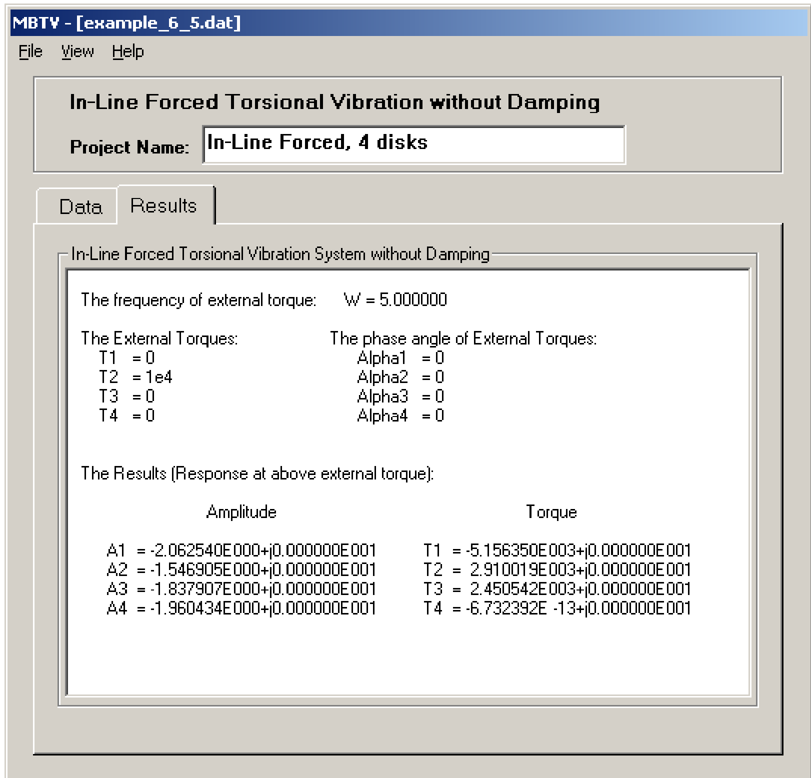

Figure 6-36 Results

■ Results Analysis

1) The results by MBTV are summarized as follows, according to Figure 6-36

ω =5.00000

The corresponding amplitudes and torques are follows:

θ_1 = -2.062540E000+j0.000000E001

θ_2 = -1.546905E000+j0.000000E001

θ_3 = -1.837907E000+j0.000000E001

T_1 = -5.156350E003+j0.000000E001

T_2 = 2.910019E003+j0.000000E001

T_3 = 2.450542E003+j0.000000E001

2) The results from Reference [43].

$$\theta_1 = -2.06\text{rad}$$

$$\theta_2 = -1.55\text{rad}$$

$$\theta_3 = -1.836\text{rad}$$

T_1 = -5150N.m

T_2 = 2930N.m

T_3 = 2460N.m

3) The relative differences

$$\theta_1 \quad \frac{|2.06254 - 2.06|}{2.06} = 0.12\%$$

$$T_1 \quad \frac{|5156.350 - 5150|}{5150} = 0.12\%$$

The relative differences are very small. The results from different calculations are very close.

- Discussion

In Figure 6-36, the A1 to A4 are respectively the amplitudes, and T1 to T3 are respectively the torques under exciting by the external torque $T_2 = 10000\sin(5 \quad t + \varphi_2) \quad (N \cdot m)$. Similarly, we can find out the designed system whether it meets engineering demands or not.

CHAPTER 7

CONTRIBUTIONS

In recent years, the complexity of reciprocating machines (in conjunction with the trend towards higher outputs) has increased to the extent that in modern engine and driven machine systems, it is vital to ensure that adverse torsional vibration conditions are avoided in all sections of the installation. This requires that adequate calculations be performed at the design stage of the system, whether for a simple straight system or a multi-engine multi-branch system, to ensure the integrity of the design.

The original and unique study has made these valuable contributions:

- Explore and discuss existing analysis theories and methods of torsional vibration systems and achieve a deeper understanding of the analysis theory and methods of the torsional vibration.

- Develop a practical method to calculate the natural frequencies and mode shapes of free torsional vibration of multi-branch systems with one or more junctions.

- Develop a more efficient and accurate method to calculate the response of free torsional vibration and forced torsional vibration of multi-branch systems, without or with damping. It is suitable to the systems with one or more junctions.

- Develop a complete software system for multi-junction multi-branch torsional vibration as a utility and enabled tool for designers. The software is easy to use, and designers need just to input the data. The software will automatically set up the motion equations, solve these motion equations and output the results with text form, T- ω curves and mode shapes. This provides an easy method to avoid critical resonance problems in torsional vibration systems, and enables designers to make rapid decisions by adjusting inertial or stiffness properties to avoid vibration problems.

- The program also has the capability for selection of shaft sizes in torsional systems.

This study has developed an efficient and accurate method and a complete program for multi-junction multi-branch torsional vibration systems. Furthermore, future works are suggested. Using the developed work, researchers can further develop a module to do stress analysis of shafts and discs while considering different types of loadings, such as step loads and shock loads. Also, fluctuating torque load should be taken into account for analysis of forced torsional vibration.

REFERENCES

[1] Reyer, J.A. and Papalambros, P.Y., Combined optimal design and control with application to an electric DC Motor, Journal of Mechanical Design, ASME, JUNE 2002, Vol.124, 183-189.

[2] Gu, Xiaoyu, Renaud, J.E., Decision-Based Collaborative Optimization, Journal of Mechanical Design, ASME, JUNE 2002, Vol.124, 1-9.

[3] Huang, Yuan Mao and Horng, Tsan-Yao, Effect of contact gears on torsional vibration branched systems, Proceedings of the 1998 ASME/JSME Joint Pressure Vessels and Piping Conference, American Society of Mechanical Engineers, San Diego, CA, USA, Jul 26-30, 1998, pp87-94.

[4] Wilson, W. K., Practical Solution of Torsional Vibration Problems, Volume One: Frequency Calculations, 3 rd Edition, John Wiley and Sons, Inc., New York, 1956.

[5] Nestorides, E. J., A Handbook on Torsional Vibration, Cambridge University Press,Cambridge, 1958, pp. 1–241.

[6] Pestel, E. C. and F. A. Leckie, Matrix Methods in Elastomechanics, McGraw-Hill BookCompany, New York, 1963, pp. 51–129.

[7] Pilkey, W. D. and P. Y. Chang, Modern Formulas for Statics and Dynamics: A Stress-and-Strain Approach, McGraw-Hill Book Company, New York, 1978, pp.118-122.

[8] Tavares, H. F. and Prodonoff, V., New Approach For Gearbox Modelling in Finite Element Analyses of Torsional Vibration of Hear-Branched Propulsion Systems, proceedings of Shock and Vibration Bulletin 56th Symposium on Shock and Vibration, Monterey, CA, USA, 1986, pp117-125.

[9] Shaikh, N., A Direct Method for Analysis of Branched Torsional Systems, American Society of Mechanical Engineers (Paper) Sep 9-12 n 73-DET-134 1973 4p

[10] Shaikh, N., "A Direct Method for Analysis of Branched Torsional Systems", Journal of Engineering for Industry, Vol.96, 1974, pp. 1006-1009.

[11] Dawson, B., and Sidwell, J. M., "The Vibrational Properties of Branched Torsional System Having One or More Branch Points", Transactions Institute of Marine Engineers, Vol. 86, Series A, Part 12, 1974, pp.239-249.

[12] Eshleman R.L., Torsional Response of Internal Combustion Engines, Journal of Engineering for Industry, May, 1974, p441-449

[13] Sankar, S., On the Torsional Vibration of Branched Systems Using Extended Transfer Matrix Method, American Society of Mechanical Engineers (Paper) Nov 27-Dec 2 n 77-WA/DE-4 1977 8p

[14] Sankar, S., On the Torsional Vibration of Branched Systems Using Extended Transfer Matrix Method, Transactions of the ASME, Journal of Mechanical Design, Vol. 101, October 1979, pp. 546-553.

[15] Mitchell, L. D., “A New Branching Technique for the Static and Dynamic Analysis of Geared Systems”, Proceedings of the Second International Conference – Vibrations in Rotating Machinery, The Institution of Mechanical Engineers, London, England, held at Cambridge, England, September 2-4, 1980, pp. 37-42.

[16] Hibner, D. H., “Dynamic Response of Viscous-Damped Multi-Shaft Jet Engines”, Journal of Aircraft, Vol. 12, No. 4, April 1975, pp. 305-312.

[17] Abhary, K., “Application of Bond Graphs In Rectilinear Modeling of Lumped Parameter Torsional Vibratory Systems”, Proceedings of the 1995 Design Engineering Technical Conferences, Volume 1: Advances in Design

Automation,ASME, New York, held at Boston, Massachusetts, September 17-20, 1995, pp. 733-740.

[18] Edwards, A. J., TORVAP-A: A Computer Program for the Torsional Vibration Analysis of Multi-Junction, Multi-Branch Systems, Computer Aided Design v 5 n 3 Jul 1973 p 139-146

[19] Gray, C.; Edwards, A. J., Torsional Vibrations of Installations With Coupled multiple engines, Transactions of Diesel Engineers & Users Association n 361 May 1974 15p

[20] Blanding, J. M., "An Analytical Study and Computer Analysis of Three-Dimensional, Steady-State Vibration of Multi-shaft Geared-Rotor Systems", Ph.D. Dissertation, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, May 1985.

[21] Tsai, J. H., and J. H. Kuang, "An Improved Branch Method for Gear-Coupled Systems", Proceedings of the 9th International Modal Analysis Conference, SEM, Bethel, Connecticut, held at Florence, Italy, April 15-18, 1992, pp.1293-1301.

[22] Doughty, S. and G. Vafaee, "Transfer Matrix Eigensolutions for Damped Torsional Systems", Transactions of the ASME, Journal of Vibration, Acoustics, Stress, and Reliability in Design, Vol. 107, January 1985, pp. 128-132.

[23] Huang, Y. M. and C. D. Horng, "Eigenvalues and Eigenvectors of Damped Branched Torsional Systems", Proceedings of the 1992 Pressure Vessels and Piping Conference: DOE Facilities Programs, Systems Interaction, and Active/Inactive Damping, ASME, New York, held at New Orleans, Louisiana, June 21-25, 1992, pp. 121-126.

[24] Mahalingam, S., Vibration of Branched Torsional Systems With Multiple Junctions, Journal of Sound and Vibration,Vol.49, No.2, 1976, pp.195-214.

[25] Mahalingam, S., Vibration of Branched System----A Displacement Excitation Approach, Journal of Sound and Vibration,Vol.46, No.2, 1976, pp.233-244.

[26] Mahalingam, S., On the Vibratory Response of Close-Coupled Systems, Journal of Sound and Vibration, Vol.63, No.2, 1979, pp.189-200.

[27] Gilbert, A. C., A note on the Calculation of Torsional Natural Frequencies of Branch System, Journal of Engineering for industry, Series B, Vol.94, No.1, Feb., 1972, pp.279-283.

[28] Hundal, M. S., An Extension of Holzer Method for Redundant Drive Trains, Journal of Engineering for Industry, Vol.96, Series B., No. 2, May 1974, pp. 697-698, 98.

[29] Dawson, B. and Davies, M., Improvement to Shaikh's Method for thr Torsional Vibration Analysis of Branched Systems, Shock and Vibration Bulletin Symp on Shock and Vib, Oct. 21-23, 1980, May 1980, San Diego, CA, USA, pp1-10.

[30] Wang, C. C., Calculating Natural Frequenies With Extended Tuplin's Method, Journal of Mechanical Design, Transactions of the ASME, Vol.103,No.2, Apr. 1981, pp379-386.

[31] Rawtani, S., Vibration Analysis of Branched System by Holzer Method, Journal of the Institution of Engineers, Part ME: Mechanical Engineering Division, Vol.63, Sep, 1982, pp64-66.

[32] Lai, J. C. S., Analysis of Discrete Vibration Systems by Transfer Matrix Method, International Journal of Mechanical Engineering Education, Vol.12, No.4, 1984, pp 253-262.

[33] Robert F.S.Jr., An Introduction to Mechanical Vibrations, Third Edition, John Wiley & Sons, 1989.

[34] Shigley, J. E. and C. R. Mischke, Mechanical Engineering Design, Fifth Edition, McGraw-Hill, Inc., New York, 1989.

[35] Thomson, W.T. and M. D. Dahleh, Theory of Vibration with Applications, Fifth Edition, Prentice Hall, Upper Saddle River, NJ, 1993, pp. 144-51.

[36] Whally, R. and Ebrahimi, M., Torsional Vibration in Rotor Shells, Proc Instn Mech Engrs, Vol.212, 1998, pp263–276.

[37] Wang, W. G., Bata, R. M. and Lyons, D. W., Analysis of torsional vibration for a multi-branch drive-train in a chassis dynamometer , Society of Automotive Engineers Transactions, Vol.100, No.Sect 2, 1991 pp629-641, Paper : 912699.

[38] Huang, Y.M., Improved extended transfer matrix method for a damped system, Proceeding of ASME Design Technical Conference, New York, USA, 1991, pp277-282.

[39] Jaksic, Nikola; Boltezar, Miha; Kuhelj, Computer simulation of torsional vibrations of branched chain topology systems , Journal of Mechanical Engineering v42 n1-2 Jan-Feb 1996, p 27-36.

[40] Mandal, B., P. Sivakumar, S. Kumar, Mandal, B., P. Sivakumar, S. Kumar, “Torsional Vibration Analysis of an Automatic Transmission", Proceedings of the IUTAM-IITD International Winter School on Optimum Dynamic Design Using Modal Testing and Structural Dynamic Modification, Allied Publishers Limited, New Delhi, held at New Delhi, December 15-19, 1997, pp. 431-444.

[41] Zhang Fengrong, The Vibration Analysis of Main Transmission System of a Steel Rolling Machine, Mechanical Science and Technologies, 1985, Vol.20(3): pp13-15,30.

[42] F.S.Tse, I.E.Morse and R.T.Hinkle, Mechanical Vibrations --- Theory and Applications, Second Edition, Allyn and Bacon, Inc., 1978.

[43] G.K.Grover, Mechanical Vibrations, Third Edition, Men Chand & Bros Roorkee, 1977.

APPENDIX

MBTV COMPUTER PROGRAM

Part 1 Mbcommon Module

In this module, there are many basic Sub programs which are used by all Form's programs.

```
variables
' Declare the variables and functions in this module
OptionExplicit
PrivateConst ROW AsInteger = 40
PrivateConst RO AsInteger = 250
Public sn As String,sm(1 To ROW) As String
Public snj As String, snsj(1 To ROW) As String
Public n AsInteger, m(1 To ROW) AsInteger
Public nj AsInteger, nsj(1 To ROW) AsInteger
Publicsinj As String,sinjj(1 To ROW) As String
PublicInj As Double, Injj(1 To ROW) As Double
Public sk(1 To ROW, 1 To ROW) As String,sinn(1 To ROW, 1 To ROW) As String
Public k(1 To ROW, 1 To ROW) As Double, Inn(1 To ROW, 1 To ROW) As Double
Public sedp(1 To ROW, 1 To ROW) As String, sidp(1 To ROW, 1 To ROW) As String
PublicEdp(1 To ROW, 1 To ROW) As Double, Idp(1 To ROW, 1 To ROW) As Double
Public sft(1 To ROW, 1 To ROW) As String, safa(1 To ROW, 1 To ROW) As String
PublicFt(1 To ROW, 1 To ROW) As String, afa(1 To ROW, 1 To ROW) As String
PublicFtr(1 To ROW, 1 To ROW) As Double,Fti(1 To ROW, 1 To ROW) As Double
Public sfft As String,Fft As Double
Publicomg3(1 To ROW) As Double
Public the3(1 To ROW) As Double, sig3(1 To ROW) As Double
Public qs AsInteger, jr AsInteger, jg AsInteger, js AsInteger
Public stt1(1 To ROW, 1 To ROW) As Double
Publicymaxs(1 To ROW) As Double, ymaxt(1 To ROW) As Double
Public ymas As Double, ymis As Double, ymat As Double, ymit As Double
Public s AsInteger, t AsInteger, u AsInteger, v AsInteger
Public z AsInteger, Fr AsInteger, Indx AsInteger
Public TextOk1 AsInteger, TextOk2 AsInteger, TextRun AsInteger
Public TextOption1 AsInteger, TextOption2 AsInteger
Public TextChang AsInteger, TextOpen AsInteger
Public PrevResults AsInteger,RunFlag AsInteger
Public sProname As String, sProdate As String,Ffname As String
Publicomg3(1 To ROW) As Double
Public the3(1 To ROW) As Double, sig3(1 To ROW) As Double
Public qs AsInteger, jr AsInteger, jg AsInteger, js AsInteger
Public stt1(1 To ROW, 1 To ROW) As Double
Publicymaxs(1 To ROW) As Double, ymaxt(1 To ROW) As Double
Public ymas As Double, ymis As Double, ymat As Double, ymit As Double
Public s AsInteger, t AsInteger, u AsInteger, v AsInteger
Public z AsInteger, Fr AsInteger, Indx AsInteger
Public TextOk1 AsInteger, TextOk2 AsInteger, TextRun AsInteger
Public TextOption1 AsInteger, TextOption2 AsInteger, TextOpen AsInteger
Public PrevResults AsInteger,RunFlag AsInteger
Public sProname, sProdate,Ffname As String
' Windows Help API Functions
Public Declare Function html help Lib "hhctrl.ocx" Alias "HtmlHelpA" (By Val hWnd As Long, by Val lpHelp
File As String, by Val wCommand As Long, by Val dwData As Long) As Long
1.1 Sub CalEnd ( )
'Calculate the deflection and residual torqueon of the end disk of the last shaft for the free tional vibration.
Public Sub CalEnd (omg As Double, thetas As Double, sigmas As Double, thetanEnd As Double, sigmanEnd
As Double)
Dim I AsInteger, j AsInteger, sgj AsInteger
Dim thetae(1 To ROW) As Double, sigmae(1 To ROW) As Double
Dim theta(1 To ROW) As Double
Dim thetans As Double, sigmans As Double
Dim thetasj As Double, sigmasj As Double
IF nj < 2 Then
```

```
sigmans =sigmas
IF n <= 2 Then
Call Holzer(omg, thetas, sigmas, thetanEnd , sigmanEnd , theta(), 1)
Exit Sub
Else
For I = 1 To n - 1
Call Holzer(omg, thetas, sigmas, thetae(i), sigmae(i), theta(), I)
Next I
Call Revise(omg, thetas, sigmas, thetae(), sigmae())
thetans =thetae(n - 1)
For I = 1 To n - 1
sigmans =sigmans +sigmae(i)
Next I
Call Holzer(omg, thetans, sigmans, thetanEnd , sigmanEnd , theta(), n)
End IF
Else
sgj = 1:thetasj =thetas:sigmasj =sigmas
For j = 1 To nj
Call Holzer(omg, thetasj, sigmasj, thetae(sgj), sigmae(sgj), theta(), sgj)
For I = sgj + 1 To sgj + nsj(j) - 2
Call Holzer(omg, thetas, sigmas, thetae(i), sigmae(i), theta(), I)
Next I
Call ReviseJ(omg, thetas, sigmas, thetae(), sigmae(), j, sgj)
thetans =thetae(sgj + nsj(j) - 2)
For I = sgj To sgj + nsj(j) - 2
sigmans =sigmans +sigmae(i)
Next I
thetasj =thetans:sigmasj =sigmans: sgj = sgj + nsj(j) - 1
Next j
End IF
End Sub
```

1.2 Sub CalEndFD( )

```
‘Calculate the deflection and residual torqueon of the end disk of the last shaft for the free tional vibration with
damping.
Public Sub CalEndFD(omg As Double, thetasr As Double, thetasi As Double, sigmasr As Double, sigmasi As
Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double)
Dim thetaer(1 To ROW) As Double, thetaei(1 To ROW) As Double
Dim sigmaer(1 To ROW) As Double, sigmaei(1 To ROW) As Double
Dim thetansr As Double, thetansi As Double
Dim sigmansr As Double, sigmansi As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim I AsInteger, I1 AsInteger
IF n < 3 Then
Call HolzerFD(omg, thetasr, thetasi, sigmasr, sigmasi, thetaer(1), thetaei(1), sigmaer(1), sigmaei(1), thetar(),
thetai(), sigmar(), sigmai(), 1)
thetaEnd r =thetaer(1):thetaEnd i =thetaei(1)
sigmaEnd r =sigmaer(1):sigmaEnd i =sigmaei(1)
Else
For I = 1 To n - 1
Call HolzerFD(omg, thetasr, thetasi, sigmasr, sigmasi, thetaer(i), thetaei(i), sigmaer(i), sigmai(i), thetar(),
thetai(), sigmar(), sigmai(), I)
Next I
Call ReviseFD(omg, thetasr, thetasi, sigmasr, sigmasi, thetaer(), thetaei(), sigmaer(), sigmaei())
thetansr =thetaer(n - 1):thetansi =thetaei(n - 1)
For I = 1 To n - 1
sigmansr =sigmansr +sigmaer(i):sigmansi =sigmaei(i)
Next I
Call HolzerFD(omg, thetasr, thetasi, sigmasr, sigmasi, thetaer(n), thetaei(n), sigmaer(n), sigmaei(n), thetar(),
thetai(), sigmar(), sigmai(), n)
thetaEnd r =thetaer(n):thetaEnd i =thetaei(n)
sigmaEnd r =sigmaer(n):sigmaEnd i =sigmaei(n)
End IF
End Sub
```

1.3 Sub CalEndFT( )

```
'Calculate the deflection and residual torqueon of the end disk of the last shaft for the forced tional vibration.
Public Sub CalEndFT(omg As Double, sigmasr() As Double, sigmasi() As Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double, ther() As Double, thei() As Double, sigr() As Double, sigi() As Double)
Dim ar As Double, br As Double, ai As Double, bi As Double
Dim Cm As Double, dm As Double, p AsInteger, q AsInteger
Dim thetasr As Double, thetasi As Double
Dim thetansr As Double, thetansi As Double
Dim sigmansr As Double, sigmansi As Double
Dim ssigr(1 To ROW, 1 To ROW) As Double, ssigi(1 To ROW, 1 To ROW) As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim thetaer(1 To ROW) As Double, thetaei(1 To ROW) As Double
Dim sigmaer(1 To ROW) As Double, sigmaei(1 To ROW) As Double
IF n < 3 Then
For q = 1 To m(1)
ssigr(1, q) = 0: ssigi(1, q) = 0
Next q
Call HolzerFT(omg, 1, 0, ssigr(), ssigi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
ar =sigmaEnd r:br =sigmaEnd i
Call HolzerFT(omg, 0, 1, ssigr(), ssigi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
ai =sigmaEnd r:bi =sigmaEnd i
Call HolzerFT(omg, 0, 0, sigmasr(), sigmasi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
Cm =sigmaEnd r: dm =sigmaEnd i
thetasr = (ai * dm -bi *Cm) / (ar *bi - ai *br)
thetasi = (ar * dm -br *Cm) / (ai *br - ar *bi)
Call HolzerFT(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
For q = 1 To m(1)
ther(1, q) =thetar(q): thei(1, q) =thetai(q)
sigr(1, q) =sigmar(q):sigi(1, q) =sigmai(q)
Next q
Else
For p = 1 To n - 1
For q = 1 To m(1)
ssigr(p, q) = 0: ssigi(p, q) = 0
Next q
Call HolzerFT(omg, 1, 0, ssigr(), ssigi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
ar =sigmaer(p):br =sigmaei(p)
Call HolzerFT(omg, 0, 1, ssigr(), ssigi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
ai =sigmaer(p):bi =sigmaei(p)
Call HolzerFT(omg, 0, 0, sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
cm =sigmaer(p): dm =sigmaei(p)
thetasr = (ai * dm -bi *Cm) / (ar *bi - ai *br)
thetasi = (ar * dm -br *Cm) / (ai *br - ar *bi)
Call HolzerFT(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
For q = 1 To m(p)
ther(p, q) =thetar(q): thei(p, q) =thetai(q)
sigr(p, q) =sigmar(q):sigi(p, q) =sigmai(q)
Next q
Next p
Call ReviseFT(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(), thetaei(), sigmaer(), sigmaei())
thetansr =thetaer(n - 1):thetansi =thetaei(n - 1)
For p = 1 To n - 1
sigmansr =sigmansr +sigmaer(p):sigmansi =sigmaei(p)
Next p
```

```
Call HolzerFT(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(n), thetaei(n), sigmaer(n), sigmaei(n), thetar(), thetai(), sigmar(), sigmai(), n)
thetaEnd r =thetaer(n):thetaEnd i =thetaei(n)
sigmaEnd r =sigmaer(n):sigmaEnd i =sigmaei(n)
For q = 1 To m(n)
ther(n, q) =thetar(q): thei(n, q) =thetai(q)
sigr(n, q) =sigmar(q):sigi(n, q) =sigmai(q)
Next q
End IF
End Sub
```

1.4 Sub CalEndFTD( )

```
'Calculate the deflection and residual torqueon of the end disk of the last shaft for the forced tional vibration with damping.
Public Sub CalEndFTD(omg As Double, sigmasr() As Double, sigmasi() As Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double, ther() As Double, thei() As Double, sigr() As Double, sigi() As Double)
Dim ar As Double, br As Double, ai As Double, bi As Double
Dim Cm As Double, dm As Double, p AsInteger, q AsInteger
Dim thetasr As Double, thetasi As Double
Dim thetansr As Double, thetansi As Double
Dim sigmansr As Double, sigmansi As Double
Dim ssigr(1 To ROW, 1 To ROW) As Double, ssigi(1 To ROW, 1 To ROW) As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim thetaer(1 To ROW) As Double, thetaei(1 To ROW) As Double
Dim sigmaer(1 To ROW) As Double, sigmaei(1 To ROW) As Double
IF n < 3 Then
For q = 1 To m(1)
ssigr(1, q) = 0: ssigi(1, q) = 0
Next q
Call HolzerFTD(omg, 1, 0, ssigr(), ssigi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
ar =sigmaEnd r:br =sigmaEnd i
Call HolzerFTD(omg, 0, 1, ssigr(), ssigi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
ai =sigmaEnd r:bi =sigmaEnd i
Call HolzerFTD(omg, 0, 0, sigmasr(), sigmasi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
cm =sigmaEnd r: dm =sigmaEnd i
thetasr = (ai * dm -bi *Cm) / (ar *bi - ai *br)
thetasi = (ar * dm -br *Cm) / (ai *br - ar *bi)
Call HolzerFTD(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, thetar(), thetai(), sigmar(), sigmai(), 1)
For q = 1 To m(1)
ther(1, q) =thetar(q): thei(1, q) =thetai(q)
sigr(1, q) =sigmar(q):sigi(1, q) =sigmai(q)
Next q
Else
For p = 1 To n - 1
For q = 1 To m(1)
ssigr(p, q) = 0: ssigi(p, q) = 0
Next q
Call HolzerFTD(omg, 1, 0, ssigr(), ssigi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
ar =sigmaer(p):br =sigmaei(p)
Call HolzerFTD(omg, 0, 1, ssigr(), ssigi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
ai =sigmaer(p):bi =sigmaei(p)
Call HolzerFTD(omg, 0, 0, sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
cm =sigmaer(p): dm =sigmaei(p)
thetasr = (ai * dm -bi *Cm) / (ar *bi - ai *br)
thetasi = (ar * dm -br *Cm) / (ai *br - ar *bi)
Call HolzerFTD(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p),
```

```
thetar(), thetai(), sigmar(), sigmai(), p)
For q = 1 To m(p)
ther(p, q) =thetar(q): thei(p, q) =thetai(q)
sigr(p, q) =sigmar(q):sigi(p, q) =sigmai(q)
Next q
Next p
Call ReviseFTD(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(), thetaei(), sigmaer(), sigmaei())
thetansr =thetaer(n - 1):thetansi =thetaei(n - 1)
For p = 1 To n - 1
sigmansr =sigmansr +sigmaer(p):sigmansi =sigmaei(p)
Next p
Call HolzerFTD(omg, thetasr, thetasi, sigmasr(), sigmasi(), thetaer(n), thetaei(n), sigmaer(n), sigmaei(n),
thetar(), thetai(), sigmar(), sigmai(), n)
thetaEnd r =thetaer(n):thetaEnd i =thetaei(n)
sigmaEnd r =sigmaer(n):sigmaEnd i =sigmaei(n)
For q = 1 To m(n)
ther(n, q) =thetar(q): thei(n, q) =thetai(q)
sigr(n, q) =sigmar(q):sigi(n, q) =sigmai(q)
Next q
End IF
End Sub
```

1.5 Sub Cpxalign( )

```
'treat the output Format of results
Public Sub Cpxalign(par As Double, pai As Double, sign1 As String, sign2 As String, xx1 As String, xx2 As
String)
Dim I1 AsInteger, I2 As Double, I3 AsInteger, I4 As Double
Dim I AsInteger
IF par = 0 Then
I1 = 1:I2 = 1
Else
I1 =Int(Log(Abs(par)) / Log(10)):I2 = 1
End IF
IF I1 = 0 Then
I2 = 1
ElseIF I1 < 0 Then
For I = 1 To Abs(i1)
I2 =I2 / 10
Next I
Else
For I = 1 ToI1
I2 =I2 * 10
Next I
End IF
par = par /I2
IF pai = 0 Then
I3 = 1:I4 = 1
Else
I3 =Int(Log(Abs(pai)) / Log(10)):I4 = 1
End IF
IF I3 = 0 Then
I4 = 1
ElseIF I3 < 0 Then
For I = 1 To Abs(i3)
I4 =I4 / 10
Next I
Else
For I = 1 ToI3
I4 =I4 * 10
Next I
End IF
pai = pai /I4
IF par > 0 Then
sign1 = "   "
Else
```

```
sign1 = " "
End IF
IF pai < 0 Then
sign2 = "-"
Else
sign2 = "+"
End IF
IF I1 < 0 Then
xx1 =Format(i1, "#00")
Else
xx1 =Format(i1, "#000")
End IF
IF I3 < 0 Then
xx2 =Format(i3, "#00")
Else
xx2 =Format(i3, "#000")
End IF
pai = Abs(pai)
End Sub
1.6 Sub CurveF( )
'calculate the graphics parameters of the free torsional vibration.
Public Sub CurveF(a1 AsInteger, a2 AsInteger, deys As Double, deyt As Double, ymas As Double, ymis As Double, ymat As Double, ymit As Double, b1 AsInteger, b2 AsInteger, dexs As Double, dext As Double, om() As Double, the()As Double, sig() As Double, units As Double, unittss Double)
Dim omgmax As Double, omgmin As Double, delta As Double
Dim yys1(1 To RO) As Double, yyt1(1 To RO) As Double
Dim yys2(1 To RO) As Double, yyt2(1 To RO) As Double
Dim I AsInteger, j AsInteger, j1 As Double, jj AsInteger
Dim I1 AsInteger, I2 As Double, I3 As Double, I4 As Double
Dim I5 AsInteger, I6 As Double, I7 As Double, I8 As Double
Dim C AsInteger, d AsInteger
IF (js - 1) <= 0 Then
Exit Sub
End IF
jj = (js - 1) / 2
IF omg3(jj) > 2500 Then
b1 = (Int(omg3(jj) / 500) + 1): dexs = 500
ElseIF omg3(jj) > 2000 Then
b1 =Int(omg3(jj) / 400) + 1: dexs = 400
ElseIF omg3(jj) > 1500 Then
b1 =Int(omg3(jj) / 300) + 1: dexs = 300
ElseIF omg3(jj) > 1000 Then
b1 =Int(omg3(jj) / 200) + 1: dexs = 200
ElseIF omg3(jj) > 500 Then
b1 =Int(omg3(jj) / 100) + 1: dexs = 100
ElseIF omg3(jj) > 100 Then
b1 =Int(omg3(jj) / 50) + 1: dexs = 50
Else
b1 =Int(omg3(jj) / 20) + 1: dexs = 20
End IF
omgmin = 0:omgmax =b1 * dexs: delta =omgmax / (b1 * 10)
IF Fr > 0 Then
omgmin = (fr - 1) * dexs: delta = dexs / (b1 * 10): dexs = dexs /b1
End IF
C = 0: d = 0
For I = 1 Tob1 * 10 + 1
om(i) =omgmin + (i - 1) * delta
Call CalEnd (om(i), 1, 0, the(i), sig(i))
IFsig(i) > 0 ThenC =C + 1: yys1(c) =sig(i)
IFsig(i) < 0 Then d = d + 1: yys2(d) =sig(i)
Next I
ymas = yys1(1): ymis = yys2(1)
For I = 2 ToC
IF yys1(i) > ymas Then ymas = yys1(i)
```

```
Next I
For I = 2 To d
IF yys2(i) < ymis Then ymis = yys2(i)
Next I
I3 =Int(Log(ymas - ymis) / Log(10))
I4 = 1
For I1 = 1 ToInt(i3)
I4 =I4 * 10
Next I1
IF I3 <= 6 Then
I2 = 10
Else
I2 = 100
End IF
deys = (Int((ymas - ymis) /I4) + 1) *I2: units =I4 /I2 / 10
ymas = ymas / units: ymis = ymis / units
s1 =Int((ymas - ymis) / deys) + 2
b2 = 0:jj = (js - 1) / 2
IF n > 2 Then
For I = 1 To n
b2 =b2 + m(i)
Next I
Else
b2 = m(n)
End IF
C = 0: d = 0
For j1 = 1 Tojj
For I = 1 Tob2
IF stt1(j1, I) > 0 Then
C =C + 1: yyt1(c) = stt1(j1, I)
Else
d = d + 1: yyt2(d) = stt1(j1, I)
End IF
Next I
Next j1
ymat = yyt1(1): ymit = yyt2(1)
For I = 2 ToC
IF yyt1(i) > ymat Then ymat = yyt1(i)
Next I
For I = 2 To d
IF yyt2(i) < ymit Then ymit = yyt2(i)
Next I
I5 = 0:I7 =Int(Log(ymat - ymit) / Log(10))
I8 = 1
IF I7 < 0 Then
I7 = -i7:I5 = 1
Else
I5 = 0
End IF
For I1 = 1 ToI7
IF I5 = 1 Then
I8 =I8 / 10
Else
I8 =I8 * 10
End IF
Next I1
IF I7 <= 6 Then
IF I5 = 1 Then
I6 = 0.1
Else
I6 = 10
End IF
Else
IF I5 = 1 Then
```

```
I6 = 0.01
Else
I6 = 100
End IF
End IF
deyt = (Int((ymat - ymit) /I8) + 1) *I6: unitt =I8 /I6 / 10
ymat = ymat / unitt: ymit = ymit / unitt
s2 =Int((ymat - ymit) / deyt) + 2
IF b2 <= 5 Then
IF n > 2 Then
b2 = (b2 - n + 2) * 2: dext = 5
Else
b2 =b2 * 2: dext = 5
End IF
Else
IF n > 2 Then
b2 =b2 - n + 2: dext = 10
Else
b2 =b2: dext = 10
End IF
End IF
End Sub
```

1.7 Sub CurveFD( )

```
'calculate the graphics parameters of the free torsional vibration with damping.
Public Sub CurveFD(a1 AsInteger, a2 AsInteger, deys As Double, deytAs Double, ymas As Double, yamt As Double, b1 AsInteger, b2 AsInteger, dexs As Double, dext As Double, units As Double, unitt As Double, om() As Double, the()As Double, sig() As Double, ther() As Double, thei()As Double, sigr() As Double, sigi() As Double)
Dim omgmax As Double, omgmin As Double, delta As Double
Dim yys1(1 To RO) As Double, yyt1(1 To RO) As Double
Dim yys2(1 To RO) As Double, yyt2(1 To RO) As Double
Dim I AsInteger, j1 AsInteger, k1 Assingle
Dim I1 AsInteger, I2 As Double, I3 AsInteger, I4 As Double
Dim I5 AsInteger, I6 As Double, I7 AsInteger, I8 As Double
Dim smsg As String, p AsInteger, q AsInteger
Dim C AsInteger, d AsInteger
Call RunFD(om(), the(), sig(), ther(), thei(), sigr(), sigi(), b1, dexs)
ymas =sig(1, n, m(n)): ymat = the(1, n, m(n))
For I = 2 Tob1 * 10
IFsig(i, n, m(n)) > ymas Then ymas =sig(i, n, m(n))
IF The(i, n, m(n)) > ymat Then ymat = the(i, n, m(n))
Next I
I3 =Int(Log(ymas) / Log(10))
I4 = 1
For I1 = 1 ToI3
I4 =I4 * 10
Next I1
IF I3 <= 6 Then
I2 = 10
Else
I2 = 100
End IF
deys = (Int(ymas /I4) + 1) *I2: units =I4 /I2 / 10
ymas = ymas / units
s1 =Int(ymas / deys) + 1
b2 = 0:j1 = (js - 1) / 2
IF n > 2 Then
For I = 1 To n
b2 =b2 + m(i)
Next I
Else
b2 = m(n)
End IF
C = 0: d = 0
```

```
For j1 = 1 Toj1
For I = 1 Tob2
IF stt1(j1, I) > 0 Then
C =C + 1: yyt1(c) = stt1(j1, I)
Else
d = d + 1: yyt2(d) = stt1(j1, I)
End IF
Next I
Next j1
ymat = yyt1(1): ymit = yyt2(1)
For I = 2 ToC
IF yyt1(i) > ymat Then ymat = yyt1(i)
Next I
For I = 2 To d
IF yyt2(i) < ymit Then ymit = yyt2(i)
Next I
I5 = 0:I7 =Int(Log(ymat - ymit) / Log(10))
I8 = 1
IF I7 < 0 Then
I7 = -i7:I5 = 1
Else
I5 = 0
End IF
For I1 = 1 ToI7
IF I5 = 1 Then
I8 =I8 / 10
Else
I8 =I8 * 10
End IF
Next I1
IF I7 <= 6 Then
IF I5 = 1 Then
I6 = 0.1
Else
I6 = 10
End IF
Else
IF I5 = 1 Then
I6 = 0.01
Else
I6 = 100
End IF
End IF
deyt = (Int((ymat - ymit) /I8) + 1) *I6: unitt =I8 /I6 / 10
ymat = ymat / unitt: ymit = ymit / unitt
s2 =Int((ymat - ymit) / deyt) + 2
IF b2 <= 5 Then
IF n > 2 Then
b2 = (b2 - n + 2) * 2: dext = 5
Else
b2 =b2 * 2: dext = 5
End IF
Else
IF n > 2 Then
b2 =b2 - n + 2: dext = 10
Else
b2 =b2: dext = 10
End IF
End IF
End Sub
```

1.8 Sub CurveFT( )

```
'calculate the graphics parameters of the forced torsional vibration.
Public Sub CurveFT(a1 AsInteger, a2 AsInteger, deys As Double, deyt As Double, ymaxs() As Double, yamxt() As Double, b AsInteger, dex As Double, om() As Double, units As Double, unitt As Double, the1() As
```

```
Double, sig1() As Double, ther1() As Double, thei1() As Double, sigr1()As Double, sigi1() As Double)
Dim ther(1 To ROW, 1 To ROW) As Double, sigr(1 To ROW, 1 To ROW) As Double
Dim thei(1 To ROW, 1 To ROW) As Double, sigi(1 To ROW, 1 To ROW) As Double
Dim omgmax As Double, omgmin As Double, delta As Double
Dim I AsInteger, jj AsInteger, k1 Assingle
Dim I1 AsInteger, I2 As Double, I3 AsInteger, I4 As Double
Dim I5 AsInteger, I6 As Double, I7 AsInteger, I8 As Double
Dim smsg As String, p AsInteger, q AsInteger
IF (js - 1) <= 0 Then
TextRun = 0:Exit Sub
End IF
jj = (js - 1) / 2
IF omg3(jj) > 2500 Then
b = (Int(omg3(jj) / 500) + 1): dex = 500
ElseIF omg3(jj) > 2000 Then
b =Int(omg3(jj) / 400) + 1: dex = 400
ElseIF omg3(jj) > 1500 Then
b =Int(omg3(jj) / 300) + 1: dex = 300
ElseIF omg3(jj) > 1000 Then
b =Int(omg3(jj) / 200) + 1: dex = 200
ElseIF omg3(jj) > 500 Then
b =Int(omg3(jj) / 100) + 1: dex = 100
ElseIF omg3(jj) > 100 Then
b =Int(omg3(jj) / 50) + 1: dex = 50
Else
b =Int(omg3(jj) / 20) + 1: dex = 20
End IF
omgmin = 1:omgmax =b * dex: delta =omgmax / (b * 10):om(1) =omgmin
Call RunFT(om(1), ther(), thei(), sigr(), sigi())
IF n < 3 Then
For I1 = 1 To m(n)
sig1(1, 1, I1) = Sqr(sigr(1, I1) ^ 2 +sigi(1, I1) ^ 2)
The1(1, 1, I1) = Sqr(ther(1, I1) ^ 2 + thei(1, I1) ^ 2)
sigr1(1, 1, I1) = Abs(sigr(1, I1))
sigi1(1, 1, I1) = Abs(sigi(1, I1))
Ther1(1, 1, I1) = Abs(ther(1, I1))
Thei1(1, 1, I1) = Abs(thei(1, I1))
ymaxs(i1) =sig1(1, 1, I1):ymaxt(i1) = the1(1, 1, I1)
Next I1
Else
For p = 1 To n
For q = 1 To m(p)
sig1(1, p, q) = Sqr(sigr(p, q) ^ 2 +sigi(p, q) ^ 2)
The1(1, p, q) = Sqr(ther(p, q) ^ 2 + thei(p, q) ^ 2)
sigr1(1, p, q) = Abs(sigr(p, q))
sigi1(1, p, q) = Abs(sigi(p, q))
Ther1(1, p, q) = Abs(ther(p, q))
Thei1(1, p, q) = Abs(thei(p, q))
ymaxs(q) =sig1(1, p, q):ymaxt(q) = the1(1, p, q)
Next q
Next p
End IF
For I = 2 Tob * 10
om(i) =omgmin + (i - 1) * delta
Call RunFT(om(i), ther(), thei(), sigr(), sigi())
IF n < 3 Then
For I1 = 1 To m(n)
sig1(i, 1, I1) = Sqr(sigr(1, I1) ^ 2 +sigi(1, I1) ^ 2)
sigr1(i, 1, I1) = Abs(sigr(1, I1)):sigi1(i, 1, I1) = Abs(sigi(1, I1))
IFsig1(i, 1, I1) >ymaxs(i1) Thenymaxs(i1) =sig1(i, 1, I1)
The1(i, 1, I1) = Sqr(ther(1, I1) ^ 2 + thei(1, I1) ^ 2)
Ther1(i, 1, I1) = Abs(ther(1, I1)): thei1(i, 1, I1) = Abs(thei(1, I1))
IF The1(i, 1, I1) >ymaxt(i1) Thenymaxt(i1) = the1(i, 1, I1)
Next I1
```

```
Else
For p = 1 To n
For q = 1 To m(p)
sig1(i, p, q) = Sqr(sigr(p, q) ^ 2 +sigi(p, q) ^ 2)
sigr1(i, p, q) = Abs(sigr(p, q))
sigi1(i, p, q) = Abs(sigi(p, q))
IFsig1(i, p, q) >ymaxs(q) Thenymaxs(q) =sig1(i, p, q)
The1(i, p, q) = Sqr(ther(p, q) ^ 2 + thei(p, q) ^ 2)
Ther1(i, p, q) = Abs(ther(p, q))
Thei1(i, p, q) = Abs(thei(p, q))
IF The1(i, p, q) >ymaxt(q) Thenymaxt(q) = the1(i, p, q)
Next q
Next p
End IF
Next I
I3 =Int(Log(ymaxs(v)) / Log(10))
I4 = 1
For I1 = 1 ToI3
I4 =I4 * 10
Next I1
IF I3 <= 6 Then
I2 = 10
Else
I2 = 100
End IF
deys = (Int(ymaxs(v) /I4) + 1) *I2: units =I4 /I2 / 10
ymaxs(v) =ymaxs(v) / units
s1 =Int(ymaxs(v) / deys) + 1
I5 = 0:I7 =Int(Log(ymaxt(v)) / Log(10))
IF I7 < 0 Then
I7 = -i7:I5 = 1
Else
I5 = 0
End IF
I8 = 1
For I1 = 1 ToI7
IF I5 = 1 Then
I8 =I8 / 10
Else
I8 =I8 * 10
End IF
Next I1
IF I7 <= 6 Then
IF I5 = 1 Then
I6 = 0.1
Else
I6 = 10
End IF
Else
IF I5 = 1 Then
I6 = 0.01
Else
I6 = 100
End IF
End IF
deyt = (Int(ymaxt(v) /I8) + 1) *I6: unitt =I8 /I6 / 10
ymaxt(v) =ymaxt(v) / unitt
s2 =Int(ymaxt(v) / deyt) + 1
End Sub
```

1.9 Sub CurveFTD( )

```
'calculate the graphics parameters of the forced torsional vibration with damping.
Public Sub CurveFTD(a1 AsInteger, a2 AsInteger, deys As Double, deyt As Double, ymaxs() As Double, yamxt() As Double, b AsInteger, dex As Double, om() As Double, units As Double, unitt As Double, the1() As Double, sig1() As Double, ther1() As Double, thei1() As Double, sigr1()As Double, sigi1() As Double)
```

```
Dim ther(1 To ROW, 1 To ROW) As Double, sigr(1 To ROW, 1 To ROW) As Double
Dim thei(1 To ROW, 1 To ROW) As Double, sigi(1 To ROW, 1 To ROW) As Double
Dim omgmax As Double, omgmin As Double, delta As Double
Dim I AsInteger, jj AsInteger, k1 Assingle, p AsInteger, q AsInteger
Dim I1 AsInteger, I2 As Double, I3 AsInteger, I4 As Double
Dim I5 AsInteger, I6 As Double, I7 AsInteger, I8 As Double
Dim smsg As String
IF (js - 1) <= 0 Then
Exit Sub
End IF
jj = (js - 1) / 2
IF omg3(jj) > 2500 Then
b = (Int(omg3(jj) / 500) + 1): dex = 500
ElseIF omg3(jj) > 2000 Then
b =Int(omg3(jj) / 400) + 1: dex = 400
ElseIF omg3(jj) > 1500 Then
b =Int(omg3(jj) / 300) + 1: dex = 300
ElseIF omg3(jj) > 1000 Then
b =Int(omg3(jj) / 200) + 1: dex = 200
ElseIF omg3(jj) > 500 Then
b =Int(omg3(jj) / 100) + 1: dex = 100
ElseIF omg3(jj) > 100 Then
b =Int(omg3(jj) / 50) + 1: dex = 50
Else
b =Int(omg3(jj) / 20) + 1: dex = 20
End IF
omgmin = 1:omgmax =b * dex: delta =omgmax / (b * 10):om(1) =omgmin
Call RunFTD(om(1), ther(), thei(), sigr(), sigi())
IF n < 3 Then
For I1 = 1 To m(n)
sig1(1, 1, I1) = Sqr(sigr(1, I1) ^ 2 +sigi(1, I1) ^ 2)
The1(1, 1, I1) = Sqr(ther(1, I1) ^ 2 + thei(1, I1) ^ 2)
sigr1(1, 1, I1) = Abs(sigr(1, I1))
sigi1(1, 1, I1) = Abs(sigi(1, I1))
Ther1(1, 1, I1) = Abs(ther(1, I1))
Thei1(1, 1, I1) = Abs(thei(1, I1))
ymaxs(i1) =sig1(1, 1, I1):ymaxt(i1) = the1(1, 1, I1)
Next I1
Else
For p = 1 To n
For q = 1 To m(p)
sig1(1, p, q) = Sqr(sigr(p, q) ^ 2 +sigi(p, q) ^ 2)
The1(1, p, q) = Sqr(ther(p, q) ^ 2 + thei(p, q) ^ 2)
sigr1(1, p, q) = Abs(sigr(1, q))
sigi1(1, p, q) = Abs(sigi(1, q))
Ther1(1, p, q) = Abs(ther(1, q))
Thei1(1, p, q) = Abs(thei(1, q))
ymaxs(i1) =sig1(1, p, q):ymaxt(i1) = the1(1, p, q)
Next q
Next p
End IF
For I = 2 Tob * 10
om(i) =omgmin + (i - 1) * delta
Call RunFTD(om(i), ther(), thei(), sigr(), sigi())
IF n < 3 Then
For I1 = 1 To m(n)
sig1(i, 1, I1) = Sqr(sigr(1, I1) ^ 2 +sigi(1, I1) ^ 2)
sigr1(i, 1, I1) = Abs(sigr(1, I1))
sigi1(i, 1, I1) = Abs(sigi(1, I1))
IFsig1(i, 1, I1) >ymaxs(i1) Thenymaxs(i1) =sig1(i, 1, I1)
The1(i, 1, I1) = Sqr(ther(1, I1) ^ 2 + thei(1, I1) ^ 2)
Ther1(i, 1, I1) = Abs(ther(1, I1))
Thei1(i, 1, I1) = Abs(thei(1, I1))
IF The1(i, 1, I1) >ymaxt(i1) Thenymaxt(i1) = the1(i, 1, I1)
```

```
Next I1
Else
For p = 1 To n
For q = 1 To m(p)
sig1(i, p, q) = Sqr(sigr(p, q) ^ 2 +sigi(p, q) ^ 2)
sigr1(i, p, q) = Abs(sigr(p, q))
sigi1(i, p, q) = Abs(sigi(p, q))
IFsig1(i, p, q) >ymaxs(q) Thenymaxs(q) =sig1(i, p, q)
The1(i, p, q) = Sqr(ther(p, q) ^ 2 + thei(p, q) ^ 2)
Ther1(i, p, q) = Abs(ther(p, q))
Thei1(i, p, q) = Abs(thei(p, q))
IF The1(i, p, q) >ymaxt(q) Thenymaxt(q) = the1(i, p, q)
Next q
Next p
End IF
Next I
I3 =Int(Log(ymaxs(v)) / Log(10))
I4 = 1
For I1 = 1 ToI3
I4 =I4 * 10
Next I1
IF I3 <= 6 Then
I2 = 10
Else
I2 = 100
End IF
deys = (Int(ymaxs(v) /I4) + 1) *I2: units =I4 /I2 / 10
ymaxs(v) =ymaxs(v) / units
s1 =Int(ymaxs(v) / deys) + 1
I5 = 0:I7 =Int(Log(ymaxt(v)) / Log(10))
IF I7 < 0 Then
I5 = 1:I7 = -i7
Else
I5 = 0
End IF
I8 = 1
For I1 = 1 ToI7
IF I5 = 1 Then
I8 =I8 / 10
Else
I8 =I8 * 10
End IF
Next I1
IF I7 <= 6 Then
IF I5 = 1 Then
I6 = 0.1
Else
I6 = 10
End IF
Else
IF I5 = 1 Then
I6 = 0.01
Else
I6 = 100
End IF
End IF
deyt = (Int(ymaxt(v) /I8) + 1) *I6: unitt =I8 /I6 / 10
ymaxt(v) =ymaxt(v) / unitt
s2 =Int(ymaxt(v) / deyt) + 1
End Sub
1.10 Sub CurXY( )
‘Make sure the X and Y Coordinate
Public Sub CurXY(xc As Double, Curx As Double, Cury As Double)
IF xc < -100000 And xc =Fix(xc) Then
```

```
Curx = -580:cury = -100
ElseIF xc <= -10000 And xc =Fix(xc) Then
Curx = -520:cury = -100
ElseIF xc <= -1000 And xc =Fix(xc) Then
Curx = -490:cury = -100
ElseIF xc <= -100 And xc =Fix(xc) Then
Curx = -380:cury = -100
ElseIF xc <= -10 And xc =Fix(xc) Then
Curx = -300:cury = -100
ElseIF xc < 1 And xc =Fix(xc) Then
Curx = -220:cury = -100
ElseIF xc < 10 And xc =Fix(xc) Then
Curx = -220:cury = -100
ElseIF xc < 100 And xc =Fix(xc) Then
Curx = -300:cury = -100
ElseIF xc < 1000 And xc =Fix(xc) Then
Curx = -380:cury = -100
ElseIF xc < 10000 And xc =Fix(xc) Then
Curx = -460:cury = -100
ElseIF xc < 1 And xc <>Fix(xc) Then
Curx = -400:cury = -100
ElseIF xc < 10 And xc <>Fix(xc) Then
Curx = -400:cury = -100
ElseIF xc < 100 And xc <>Fix(xc) Then
Curx = -420:cury = -100
ElseIF xc < 1000 And xc <>Fix(xc) Then
Curx = -460:cury = -100
Else
Curx = -520:cury = -100
End IF
End Sub
```

1.11 Sub Holzer( )

```
'Use Holzer method to calculate the deflection and the torque of a free torsional vibration system.
Public Sub Holzer(omg As Double, thetas As Double, sigmas As Double, thetaEnd As Double, sigmaEnd As Double, theta() As Double, p AsInteger)
Dim sigma(1 To ROW) As Double, q As Integer, I As Integer
IF p = n Then
IF n = 1 Then
theta(1) =thetas
sigma(1) =Inn(1, 1) *omg ^ 2 *thetas
IF m(p) = 2 Then
theta(2) =theta(1) -sigma(1) / k(p, 1)
sigma(2) =sigma(1) +Inn(p, 2) *omg ^ 2 *theta(2)
thetaEnd =theta(2):sigmaEnd =sigma(2)
Else
For q = 2 To m(p)
theta(q) =theta(q - 1) -sigma(q - 1) / k(p, q - 1)
sigma(q) =sigma(q - 1) +Inn(p, q) *omg ^ 2 *theta(q)
Next q
thetaEnd =theta(m(p)):sigmaEnd =sigma(m(p))
End IF
Else
theta(1) =thetas
sigma(1) =sigmas +Inj *omg ^ 2 *theta(1)
IF m(p) = 2 Then
theta(2) =theta(1) -sigma(1) / k(p, 1)
sigma(2) =sigma(1) +Inn(p, 2) *omg ^ 2 *theta(2)
thetaEnd =theta(2):sigmaEnd =sigma(2)
Else
For q = 2 To m(p)
theta(q) =theta(q - 1) -sigma(q - 1) / k(p, q - 1)
sigma(q) =sigma(q - 1) +Inn(p, q) *omg ^ 2 *theta(q)
Next q
thetaEnd =theta(m(p)):sigmaEnd =sigma(m(p))
```

```
End IF
End IF
Else
theta(1) =thetas
sigma(1) =Inn(p, 1) *omg ^ 2 *theta(1)
IF m(p) <= 1 Then
thetaEnd =theta(1) -sigma(1) / k(p, 1)
sigmaEnd =sigma(1)
ElseIF m(p) = 2 Then
theta(2) =theta(1) -sigma(1) / k(p, 1)
sigma(2) =sigma(1) +Inn(p, 2) *omg ^ 2 *theta(2)
thetaEnd =theta(2) -sigma(2) / k(p, 2)
sigmaEnd =sigma(2)
Else
For q = 2 To m(p)
theta(q) =theta(q - 1) -sigma(q - 1) / k(p, q - 1)
sigma(q) =sigma(q - 1) +Inn(p, q) *omg ^ 2 *theta(q)
Next q
thetaEnd =theta(m(p)) -sigma(m(p)) / k(p, m(p))
sigmaEnd =sigma(m(p))
End IF
End IF
End Sub
```

1.12 Sub HolzerFD( )

```
'Use Holzer method to calculate the deflection and the torque of a free torsional vibration system with damping.
Public Sub HolzerFD(omg As Double, thetasr As Double, thetasi As Double, sigmasr As Double, sigmasi As Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double, thetar()As Double, thetai() As Double, sigmar() As Double, sigmai()As Double, p AsInteger)
Dim q AsInteger, I AsInteger
IF p = n Then
IF n = 1 Then
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =Inn(1, 1) *omg ^ 2 *thetasr +omg *thetai(1) *Edp(1, 1)
sigmai(1) =Inn(1, 1) *omg ^ 2 *thetasi -omg *thetar(1) *Edp(1, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) -omg *Edp(p, 1) *thetar(2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
sigmaEnd i =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q -1) ^ 2 +omg ^ 2 *idp(p, q - 1) ^ 2)
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2 *idp(p, q - 1) ^ 2)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q)sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg*Edp(p, q) *thetar(q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
Else      "n>1"
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr +Inj *omg ^ 2 *thetasr +omg *thetai(1) *Edp(1, 1)
sigmai(1) =sigmasi +Inj *omg ^ 2 *thetasi -omg *thetar(1) *Edp(1, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) -omg *Edp(p, 1) *thetar(2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
```

```
sigmaEnd i =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*idp(p, q - 1) ^ 2)
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) ^ 2)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg *Edp(p, q) *thetar(q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
Else
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =Inn(p, 1) *omg ^ 2 *thetasr +omg *thetai(1) *Edp(p, 1)
sigmai(1) =Inn(p, 1) *omg ^ 2 *thetasi -omg *thetar(1) *Edp(p, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) -omg *Edp(p, 1) *thetar(2)
thetaEnd r =thetar(2) - (sigmar(2) * k(p, 2) +omg *Idp(p, 2) *sigmai(2)) / (k(p, 2) ^ 2 +omg ^ 2 *Idp(p, 2) ^ 2)
thetaEnd i =thetai(2) - (sigmai(2) * k(p, 2) -omg *Idp(p, 2) *sigmar(2)) / (k(p, 2) ^ 2 +omg ^ 2 *Idp(p, 2) ^ 2)
sigmaEnd i =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) ^ 2)
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*idp(p, q - 1) ^ 2)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg *Edp(p, q) *thetar(q)
Next q
thetaEnd r =thetar(m(p)) - (sigmar(m(p)) * k(p, m(p)) +omg *Idp(p, m(p)) *sigmai(m(p))) / (k(p, m(p)) ^ 2 +omg
^ 2 *Idp(p, m(p)) ^ 2)
thetaEnd i =thetai(m(p)) - (sigmai(m(p)) * k(p, m(p)) -omg *idp(p, m(p)) *sigmar(m(p))) / (k(p, m(p)) ^ 2 +omg ^
2 *idp(p, m(p)) ^ 2)
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
End Sub
```

1.13 Sub HolzerFT( )

```
'Use Holzer method to calculate the deflection and the torque of a forced torsional vibration system.
Public Sub HolzerFT(omg As Double, thetasr As Double, thetasi As Double, sigmasr()As Double, sigmasi() As Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double, thetar() As Double, thetai()As Double, sigmar() As Double, sigmai() As Double, p AsInteger)
Dim q AsInteger
IF p = n Then
IF n = 1 Then
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inn(1, 1) *omg ^ 2 *thetar(1) +sigmasr(1, 1)
sigmai(1) =sigmasi(p, 1) +Inn(1, 1) *omg ^ 2 *thetai(1) +sigmasi(1, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) -sigmar(1) / k(p, 1)
thetai(2) =thetai(1) -sigmai(1) / k(p, 1)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(2) +sigmasi(p, 2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) -sigmar(q - 1) / k(p, q - 1)
```

```
thetai(q) =thetai(q - 1) -sigmai(q - 1) / k(p, q - 1)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) +sigmasi(p, q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
Else     '''n>3'''
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inj *omg ^ 2 *thetar(1)
sigmai(1) =sigmasi(p, 1) +Inj *omg ^ 2 *thetai(1)
IF m(p) = 2 Then
thetar(2) =thetar(1) -sigmar(1) / k(p, 1)
thetai(2) =thetai(1) -sigmai(1) / k(p, 1)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) +sigmasi(p, 2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) -sigmar(q - 1) / k(p, q - 1)
thetai(q) =thetai(q - 1) -sigmai(q - 1) / k(p, q - 1)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) _+sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) +sigmasi(p, q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
Else      ''''p<n''''
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inn(p, 1) *omg ^ 2 *thetar(1)
sigmai(1) =sigmasi(p, 1) +Inn(p, 1) *omg ^ 2 *thetai(1)
IF m(p) = 2 Then
thetar(2) =thetar(1) -sigmar(1) / k(p, 1)
thetai(2) =thetai(1) -sigmai(1) / k(p, 1)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) +sigmasi(p, 2)
thetaEnd r =thetar(2) -sigmar(2) / k(p, 2)
thetaEnd i =thetai(2) -sigmai(2) / k(p, 2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) -sigmar(q - 1) / k(p, q - 1)
thetai(q) =thetai(q - 1) -sigmai(q - 1) / k(p, q - 1)
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) +sigmasi(p, q)
Next q
thetaEnd r =thetar(q) -sigmar(q) / k(p, q)
thetaEnd i =thetai(q) -sigmai(q) / k(p, q)
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
End Sub
```

1.14 Sub HolzerFTD( )

```
' Use Holzer method to calculate the deflection and the torque of a forced torsional vibration system.
Public Sub HolzerFTD(omg As Double, thetasr As Double, thetasi As Double, sigmasr() As Double, sigmasi() As Double, thetaEnd r As Double, thetaEnd i As Double, sigmaEnd r As Double, sigmaEnd i As Double, thetar() As Double, thetai()As Double, sigmar() As Double, sigmai() As Double, p AsInteger)
Dim q AsInteger
IF p = n Then
IF n = 1 Then
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inn(1, 1) *omg ^ 2 *thetar(1) +omg *thetai(1) *Edp(1, 1)
```

```
sigmai(1) =sigmasi(p, 1) +Inn(1, 1) *omg ^ 2 *thetai(1) -omg *thetar(1) *Edp(1, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(2) -omg *Edp(p, 1) *thetar(2) +sigmasi(p, 2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q) +sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg *Edp(p, q) *thetar(q) +sigmasi(p, q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
Else     '''n>3'''
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inj *omg ^ 2 *thetar(1) +omg *thetai(1) *Edp(1, 1)
sigmai(1) =sigmasi(p, 1) +Inj *omg ^ 2 *thetai(1) -omg *thetar(1) *Edp(1, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) -omg *Edp(p, 1) *thetar(2) +sigmasi(p, 2)
thetaEnd r =thetar(2):thetaEnd i =thetai(2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q) +sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg *Edp(p, q) *thetar(q) +sigmasi(p, q)
Next q
thetaEnd r =thetar(m(p)):thetaEnd i =thetai(m(p))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
Else      ''''p<n''''
thetar(1) =thetasr:thetai(1) =thetasi
sigmar(1) =sigmasr(p, 1) +Inn(p, 1) *omg ^ 2 *thetar(1) +omg *thetai(1) *Edp(p, 1)
sigmai(1) =sigmasi(p, 1) +Inn(p, 1) *omg ^ 2 *thetasi -omg *thetar(1) *Edp(p, 1)
IF m(p) = 2 Then
thetar(2) =thetar(1) - (sigmar(1) * k(p, 1) +omg *Idp(p, 1) *sigmai(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
thetai(2) =thetai(1) - (sigmai(1) * k(p, 1) -omg *Idp(p, 1) *sigmar(1)) / (k(p, 1) ^ 2 +omg ^ 2 *Idp(p, 1) ^ 2)
sigmar(2) =sigmar(1) +Inn(p, 2) *omg ^ 2 *thetar(2) +omg *Edp(p, 2) *thetai(2) +sigmasr(p, 2)
sigmai(2) =sigmai(1) +Inn(p, 2) *omg ^ 2 *thetai(1) -omg *Edp(p, 1) *thetar(2) +sigmasi(p, 2)
thetaEnd r =thetar(2) - (sigmar(2) * k(p, 2) +omg *Idp(p, 2) *sigmai(2)) / (k(p, 2) ^ 2 +omg ^ 2 *idp(p, 2) ^ 2)
thetaEnd i =thetai(2) - (sigmai(2) * k(p, 2) -omg *Idp(p, 2) *sigmar(2)) / (k(p, 2) ^ 2 +omg ^ 2 *Idp(p, 2) ^ 2)
sigmaEnd r =sigmar(2):sigmaEnd i =sigmai(2)
Else
For q = 2 To m(p)
thetar(q) =thetar(q - 1) - (sigmar(q - 1) * k(p, q - 1) +omg *Idp(p, q - 1) *sigmai(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
thetai(q) =thetai(q - 1) - (sigmai(q - 1) * k(p, q - 1) -omg *Idp(p, q - 1) *sigmar(q - 1)) / (k(p, q - 1) ^ 2 +omg ^ 2
*Idp(p, q - 1) *Idp(p, q - 1))
sigmar(q) =sigmar(q - 1) +Inn(p, q) *omg ^ 2 *thetar(q) +omg *Edp(p, q) *thetai(q) +sigmasr(p, q)
sigmai(q) =sigmai(q - 1) +Inn(p, q) *omg ^ 2 *thetai(q) -omg *Edp(p, q) *thetar(q) +sigmasi(p, q)
```

```
Next q
thetaEnd r =thetar(q) - (sigmar(q) * k(p, q) +omg *Idp(p, q) *sigmai(q)) / (k(p, q) ^ 2 +omg ^ 2 *Idp(p, q) *Idp(p,
q))
thetaEnd i =thetai(q) - (sigmai(q) * k(p, q) -omg *Idp(p, q) *sigmar(q)) / (k(p, q) ^ 2 +omg ^ 2 *Idp(p, q) *Idp(p,
q))
sigmaEnd r =sigmar(m(p)):sigmaEnd i =sigmai(m(p))
End IF
End IF
End Sub
```

1.15 Sub MBoption1( )

```
'Draw the sketch of In-Line torsional vibration system
Public Sub MBoption1()
Dim X As Long, dex AsInteger, x1 AsInteger, y1 AsInteger
x1 = 180: y1 = 0
x = RGB(0, 0, 0)
WithMBTV.select
.AutoRedraw = True
.Label8(0).Caption = "":.Label8(1).Caption = "":.Label8(2).Caption = ""
.Label8(3).Caption = "":.Label8(4).Caption = "":.Label8(5).Caption = ""
.Label8(6).Caption = "":.Label9.Caption = ""
.Label8(0).Left = 0:.Label8(0).Top = 0
.Label8(1).Left = 0:.Label8(1).Top = 0
.Label8(2).Left = 0:.Label8(2).Top = 0
.Label8(3).Left = 0:.Label8(3).Top = 0
.Label8(4).Left = 0:.Label8(4).Top = 0
.Label8(5).Left = 0:.Label8(5).Top = 0
.Label8(6).Left = 0:.Label8(6).Top = 0
.Label9.Left = 0:.Label9.Top = 0
.Picture1.Picture = LoadPicture()
.Label8(0).Left = 330 + x1:.Label8(0).Top = 2420
.Label8(0).Caption = "1"
.Label8(1).Left = 860 + x1:.Label8(1).Top = 2420
.Label8(1).Caption = "2"
.Label8(2).Left = 4810 + x1:.Label8(2).Top = 2420
.Label8(2).Caption = "n-1"
.Label8(3).Left = 5480 + x1:.Label8(3).Top = 2420
.Label8(3).Caption = "n"
.Picture1.DrawWidth = 1
.Picture1.Line (365 + x1, 1890)-(5510 + x1, 1890), X
.Picture1.Line (2105 + x1, 1300)-(2935 + x1, 1300), X
.Picture1.Line (2830 + x1, 1205)-(2935 + x1, 1310), X
.Picture1.Line (2830 + x1, 1415)-(2935 + x1, 1310), X
.Picture1.DrawWidth = 4
.Picture1.Line (365 + x1, 1470)-(365 + x1, 2310), X
.Picture1.Line (1730 + x1, 1365)-(1730 + x1, 2415), X
.Picture1.Line (3935 + x1, 1365)-(3935 + x1, 2415), X
.Picture1.Line (5510 + x1, 1470)-(5510 + x1, 2310), X
.Picture1.DrawWidth = 3
.Picture1.Line (890 + x1, 1575)-(890 + x1, 2205), X
.Picture1.Line (2465 + x1, 1470)-(2465 + x1, 2310), X
.Picture1.Line (3200 + x1, 1470)-(3200 + x1, 2310), X
.Picture1.Line (4880 + x1, 1515)-(4880 + x1, 2205), X
dex = TextOption2
selectCase dex
Case 5
.Label9.Left = 1800:.Label9.Top = 600
.Label9.Caption = "In-LineFree Torsional Vibration System"
Case 6
.Label9.Left = 1300:.Label9.Top = 600
.Label9.Caption = "In-LineFree Torsional Vibration System with Damping"
Case 7
.Label9.Left = 1250:.Label9.Top = 600
.Label9.Caption = "In-LineFor ced Torsional Vibration System with Damping"
Case 8
```

```
.Label9.Left = 1050:.Label9.Top = 600
.Label9.Caption = "In-LineFor ced Torsional Vibration System without Damping"
Case 0
.Label9.Caption = ""
End Select
.AutoRedraw =False
End With
End Sub
```

1.16 Sub MBoption2( )

```
'Draw the sketchof multi-branch torsional vibration system
Public Sub MBoption2()
Dim X As Long, dex AsInteger, x1 AsInteger, y1 AsInteger
x1 = 170: y1 = 0
x = RGB(0, 0, 0)
WithMBTV.select
.AutoRedraw = True
.Label8(0).Caption = "":.Label8(1).Caption = "":.Label8(2).Caption = ""
.Label8(3).Caption = "":.Label8(4).Caption = "":.Label8(5).Caption = ""
.Label8(6).Caption = "":.Label9.Caption = ""
.Label8(0).Left = 0:.Label8(0).Top = 0
.Label8(1).Left = 0:.Label8(1).Top = 0
.Label8(2).Left = 0:.Label8(2).Top = 0
.Label8(3).Left = 0:.Label8(3).Top = 0
.Label8(4).Left = 0:.Label8(4).Top = 0
.Label8(5).Left = 0:.Label8(5).Top = 0
.Label8(6).Left = 0:.Label8(6).Top = 0
.Label9.Left = 0:.Label9.Top = 0
.Picture1.Picture = LoadPicture()
.Label8(0).Left = 2050 + x1:.Label8(0).Top = 830
.Label8(0).Caption = "Branch 1"
.Label8(1).Left = 3550 + x1:.Label8(1).Top = 400
.Label8(1).Caption = "Branch 2"
.Label8(2).Left = 4850 + x1:.Label8(2).Top = 2085
.Label8(2).Caption = "Branch (n-1)"
.Label8(3).Left = 3450 + x1:.Label8(3).Top = 2760
.Label8(3).Caption = "Branch n"
.Picture1.DrawWidth = 1
.Picture1.Line (315 + x1, 1690)-(5460 + x1, 1690), X
.Picture1.Line (3885 + x1, 1690)-(4935 + x1, 640), X
.Picture1.Line (3885 + x1, 1690)-(4935 + x1, 2740), X
.Picture1.Line (3995 + x1, 845)-(4415 + x1, 425), X
.Picture1.Line (3995 + x1, 845)-(3995 + x1, 740), X
.Picture1.Line (3995 + x1, 845)-(4100 + x1, 845), X
.Picture1.Line (3985 + x1, 2490)-(4410 + x1, 2915), X
.Picture1.Line (4410 + x1, 2810)-(4410 + x1, 2915), X
.Picture1.Line (4305 + x1, 2915)-(4410 + x1, 2915), X
.Picture1.Line (5300 + x1, 2350)-(5825 + x1, 2350), X
.Picture1.Line (5300 + x1, 2350)-(5405 + x1, 2245), X
.Picture1.Line (5300 + x1, 2350)-(5405 + x1, 2455), X
.Picture1.Line (1995 + x1, 1110)-(2835 + x1, 1110), X
.Picture1.Line (2730 + x1, 1005)-(2835 + x1, 1110), X
.Picture1.Line (2730 + x1, 1215)-(2835 + x1, 1110), X
.Picture1.DrawWidth = 4
.Picture1.Line (315 + x1, 1270)-(315 + x1, 2110), X
.Picture1.Line (1680 + x1, 1065)-(1680 + x1, 2215), X
.Picture1.Line (3885 + x1, 1265)-(3885 + x1, 2215), X
.Picture1.Line (5460 + x1, 1170)-(5460 + x1, 2110), X
.Picture1.Line (4620 + x1, 325)-(5250 + x1, 955), X
.Picture1.Line (4620 + x1, 3055)-(5250 + x1, 2425), X
.Picture1.DrawWidth = 3
.Picture1.Line (840 + x1, 1375)-(840 + x1, 2005), X
.Picture1.Line (2415 + x1, 1170)-(2415 + x1, 2110), X
.Picture1.Line (3150 + x1, 1270)-(3150 + x1, 2110), X
.Picture1.Line (4830 + x1, 1215)-(4830 + x1, 2005), X
```

```
.Picture1.Line (4320 + x1, 1000)-(4635 + x1, 1315), X
.Picture1.Line (4290 + x1, 2410)-(4605 + x1, 2095), X
dex = TextOption2
selectCase dex
Case 5
.Label9.Left = 1400:.Label9.Top = 0
.Label9.Caption = "Multi-BranchFree Torsional Vibration System"
Case 6
.Label9.Left = 1000:.Label9.Top = 0
.Label9.Caption = "Multi-BranchFree Torsional Vibration system with Damping"
Case 7
.Label9.Left = 1000:.Label9.Top = 0
.Label9.Caption = "Multi-BranchFor ced Torsional Vibration System with Damping"
Case 8
.Label9.Left = 850:.Label9.Top = 0
.Label9.Caption = "Multi-BranchFor ced Torsional Vibration System without Damping"
Case 0
.Label9.Caption = ""
End Select
.AutoRedraw =False
End With
End Sub
```

1.17 Sub MBoption3( )

```
'Draw the sketchof multi-junction torsional vibration system
Public Sub MBoption3()
Dim X As Long, dex AsInteger, x1 AsInteger, y1 AsInteger
MBTV.select.AutoRedraw = True
MBTV.select.Picture1.Picture = LoadPicture()
x1 = 120: y1 = 0
x = RGB(0, 0, 0)
WithMBTV.select
.Label8(0).Caption = "":.Label8(1).Caption = "":.Label8(2).Caption = ""
.Label8(3).Caption = "":.Label8(4).Caption = "":.Label8(5).Caption = ""
.Label8(6).Caption = "":.Label9.Caption = ""
.Label8(0).Left = 0:.Label8(0).Top = 0
.Label8(1).Left = 0:.Label8(1).Top = 0
.Label8(2).Left = 0:.Label8(2).Top = 0
.Label8(3).Left = 0:.Label8(3).Top = 0
.Label8(4).Left = 0:.Label8(4).Top = 0
.Label8(5).Left = 0:.Label8(5).Top = 0
.Label8(6).Left = 0:.Label8(6).Top = 0
.Label9.Left = 0:.Label9.Top = 0
.Label8(0).Left = 150 + x1:.Label8(0).Top = 2080
.Label8(0).Caption = "BranchI"
.Label8(1).Left = 1400 + x1:.Label8(1).Top = 400
.Label8(1).Caption = "Branch 1"
.Label8(2).Left = 2150 + x1:.Label8(2).Top = 950
.Label8(2).Caption = "Branch n"
.Label8(3).Left = 3900 + x1:.Label8(3).Top = 400
.Label8(3).Caption = "Branch (n+1)"
.Label8(4).Left = 4950 + x1:.Label8(4).Top = 2135
.Label8(4).Caption = "Branch (n+j)"
.Label8(5).Left = 3600 + x1:.Label8(5).Top = 2860
.Label8(5).Caption = "Branch m"
.Label8(6).Left = 1350 + x1:.Label8(6).Top = 2900
.Label8(6).Caption = "Branch (n-1)"
.Picture1.DrawWidth = 1
.Picture1.Line (x1 + 415, 1690)-(x1 + 5560, 1690), X
.Picture1.Line (x1 + 730, 640)-(x1 + 1780, 1690), X
.Picture1.Line (x1 + 730, 2740)-(x1 + 1780, 1690), X
.Picture1.Line (x1 + 3985, 1690)-(x1 + 5035, 640), X
.Picture1.Line (x1 + 3985, 1690)-(x1 + 5035, 2740), X
.Picture1.Line (x1 + 1150, 420)-(x1 + 1675, 945), X
.Picture1.Line (x1 + 1675, 840)-(x1 + 1675, 945), X
```

```
.Picture1.Line (x1 + 1675, 945)-(x1 + 1570, 945), X
.Picture1.Line (x1 + 1150, 3060)-(x1 + 1675, 2535), X
.Picture1.Line (x1 + 1570, 2535)-(x1 + 1675, 2535), X
.Picture1.Line (x1 + 1675, 2535)-(x1 + 1675, 2640), X
.Picture1.Line (x1 + 4195, 945)-(x1 + 4615, 525), X
.Picture1.Line (x1 + 4195, 945)-(x1 + 4195, 840), X
.Picture1.Line (x1 + 4195, 945)-(x1 + 4300, 945), X
.Picture1.Line (x1 + 3985, 2540)-(x1 + 4510, 3065), X
.Picture1.Line (x1 + 4510, 2960)-(x1 + 4510, 3065), X
.Picture1.Line (x1 + 4405, 3065)-(x1 + 4510, 3065), X
.Picture1.Line (x1 + 2095, 1210)-(x1 + 2935, 1210), X
.Picture1.Line (x1 + 2830, 1150)-(x1 + 2935, 1210), X
.Picture1.Line (x1 + 2830, 1315)-(x1 + 2935, 1210), X
.Picture1.Line (x1 + 205, 2350)-(x1 + 835, 2350), X
.Picture1.Line (x1 + 730, 2245)-(x1 + 835, 2350), X
.Picture1.Line (x1 + 730, 2455)-(x1 + 835, 2350), X
.Picture1.Line (x1 + 5140, 2380)-(x1 + 5665, 2380), X
.Picture1.Line (x1 + 5140, 2380)-(x1 + 5245, 2275), X
.Picture1.Line (x1 + 5140, 2380)-(x1 + 5245, 2485), X
.Picture1.DrawWidth = 4
.Picture1.Line (x1 + 415, 1270)-(x1 + 415, 2110), X
.Picture1.Line (x1 + 1780, 1165)-(x1 + 1780, 2215), X
.Picture1.Line (x1 + 3985, 1165)-(x1 + 3985, 2215), X
.Picture1.Line (x1 + 5560, 1270)-(x1 + 5560, 2110), X
.Picture1.Line (x1 + 415, 955)-(x1 + 1045, 325), X
.Picture1.Line (x1 + 415, 2425)-(x1 + 1045, 3055), X
.Picture1.Line (x1 + 4720, 325)-(x1 + 5350, 955), X
.Picture1.Line (x1 + 4720, 3055)-(x1 + 5350, 2425), X
.Picture1.DrawWidth = 3
.Picture1.Line (x1 + 940, 1375)-(x1 + 940, 2005), X
.Picture1.Line (x1 + 2515, 1270)-(x1 + 2515, 2110), X
.Picture1.Line (x1 + 3250, 1270)-(x1 + 3250, 2110), X
.Picture1.Line (x1 + 4930, 1315)-(x1 + 4930, 2005), X
.Picture1.Line (x1 + 1060, 1285)-(x1 + 1375, 970), X
.Picture1.Line (x1 + 1060, 2095)-(x1 + 1375, 2410), X
.Picture1.Line (x1 + 4420, 1000)-(x1 + 4735, 1315), X
.Picture1.Line (x1 + 4390, 2410)-(x1 + 4705, 2095), X
dex = TextOption2
selectCase dex
Case 5
.Label9.Left = 1400:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionFree Torsional Vibration System"
Case 6
.Label9.Left = 1000:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionFree Torsional Vibration system with Damping"
Case 7
.Label9.Left = 900:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionForced Torsional Vibration System with Damping"
Case 8
.Label9.Left = 800:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionForced Torsional Vibration System without Damping"
Case 0
.Label9.Caption = ""
End Select
.AutoRedraw =False
End With
End Sub
```

1.18 Sub MBoption4( )

```
Public Sub MBoption4()
WithMBTV.select
.Label9.Caption = "":.Label9.Left = 0:.Label9.Top = 0
IF TextOption1 = 1 Then
.Label9.Left = 1800:.Label9.Top = 600
.Label9.Caption = "In-Line Free Torsional Vibration System"
```

```
ElseIF TextOption1 = 2 Then
.Label9.Left = 1700:.Label9.Top = 0
.Label9.Caption = "Multi-Branch Free Torsional Vibration system"
ElseIF TextOption1 = 3 Then
.Label9.Left = 1450:.Label9.Top = 0
.Label9.Caption = "Multi-Junction Free Torsional Vibration System"
Else
.Label9.Caption = ""
End IF
End With
End Sub
1.19 Sub MBoption5( )
 'Draw the sketch of free torsional vibration system
Public Sub MBoption5()
WithMBTV.select
.Label9.Caption = "":.Label9.Left = 0:.Label9.Top = 0
IF TextOption1 = 1 Then
.Label9.Left = 1350:.Label9.Top = 600
.Label9.Caption = "In-LineFree Torsional Vibration System with Damping"
ElseIF TextOption1 = 2 Then
.Label9.Left = 1100:.Label9.Top = “”
.Label9.Caption = "Multi-BranchFree Torsional Vibrationsystem with Damping"
ElseIF TextOption1 = 3 Then
.Label9.Left = 1100:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionFree Torsional Vibration System with Damping"
Else
.Label9.Caption = ""
End IF
End With
End Sub
1.20 Sub MBoption6( )
'Draw the sketch of free torsional vibration system with damping
Public Sub MBoption6()
WithMBTV.select
.Label9.Caption = "":.Label9.Left = 0:.Label9.Top = 0
IF TextOption1 = 1 Then
.Label9.Left = 1250:.Label9.Top = 600
.Label9.Caption = "In-LineFor ced Torsional Vibration System with Damping"
ElseIF TextOption1 = 2 Then
.Label9.Left = 1000:.Label9.Top = 0
.Label9.Caption = "Multi-branchFor ced Torsional Vibration System with Damping"
ElseIF TextOption1 = 3 Then
.Label9.Left = 950:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionFor ced Torsional Vibration System with Damping"
Else
.Label9.Caption = ""
End IF
End With
End Sub
1.21 Sub MBoption7( )
'Draw the sketch of forced torsional vibration system with damping
Public Sub MBoption7()
WithMBTV.select
.Label9.Caption = "":.Label9.Left = 0:.Label9.Top = 0
IF TextOption1 = 1 Then
.Label9.Left = 1100:.Label9.Top = 600
.Label9.Caption = "In-LineFor ced Torsional Vibration System without Damping"
ElseIF TextOption1 = 2 Then
.Label9.Left = 850:.Label9.Top = 0
.Label9.Caption = "Multi-BranchFor ced Torsional Vibration System without Damping"
ElseIF TextOption1 = 3 Then
.Label9.Left = 800:.Label9.Top = 0
.Label9.Caption = "Multi-JunctionFor ced Torsional Vibration System without Damping"
Else
```

```
.Label9.Caption = ""
End IF
End With
End Sub
```

1.22 Sub ProjectOpen( )

'the subroute for opening a existent project file, and read all data and results in the file, and put them to the interface.

```
Public Sub ProjectOpen(data For m As For m, lis As ListBox, picb As pictureBox)
Dim ln,smsg As String, anatype As Integer
Dim fname As String,Ftitle As String
OpenEr = 0
Opentype:onError GoToopenError
MBTV.select.CommonDialog1.CancelError = True
MBTV.select.CommonDialog1.ShowOpen
fname = MBTV.select.CommonDialog1.FileName
ftitle = MBTV.select.CommonDialog1.FileTitle
IF fname = "" ThenExit Sub
Open fname For Input As #1
LineInput #1, ln
LineInput #1, ln
Close
'Read the data of "In-Line" and "Free", and put them to the FORM ildata1.
IF ln = "Analysis Type:In-Line Free Torsional Vibration" Then
IF data Form.Name = "ildata1" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "    'In-Line Free Torsional Vibration'." &Chr(13) &Chr(13) & "Continue or not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.ildata1.Show: MBTV.ildata1.SSTab1 = 0
Call MBTV.ildata1.openildata1(fname)
MBTV.ildata1.Caption = "MBTV - [" &Ftitle & "]"
MBTV.ildata1.saveproject.Enabled = True
MBTV.ildata1.SaveAs.Enabled = True
MBTV.ildata1.printproject.Enabled = True
MBTV.ildata1.preresults.Enabled = True
MBTV.ildata1.viewdata.Enabled = True
MBTV.ildata1.viewresult.Enabled = True
MBTV.ildata1.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
'Read the data of "In-Line" and "Free with Damping", and put them to the FORM ildata2.
Else IF ln = "Analysis Type:In-Line Torsional Vibration with Damping" Then
IF dataForm.Name = "ildata2" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "    'In-Line Torsional Vibration with Damping'." &Chr(13) &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
```

```
MBTV.ildata2.Show: MBTV.ildata2.SSTab1 = 0
Call MBTV.ildata2.openildata2(fname)
MBTV.ildata2.Caption = "MBTV - [" &Ftitle & "]"
MBTV.ildata2.saveproject.Enabled = True
MBTV.ildata2.SaveAs.Enabled = True
MBTV.ildata2.printproject.Enabled = True
MBTV.ildata2.preresults.Enabled = True
MBTV.ildata2.viewdata.Enabled = True
MBTV.ildata2.viewresult.Enabled = True
MBTV.ildata2.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “In-Line” and “Forced with Damping”, and put them to the FORM ildata3.
Else IF In = "Analysis Type:In-Line Forced Torsional Vibration with Damping" Then
IF dataForm.Name = "ildata3" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis typeof the File which will be opened Is:" &Chr(13)
& " 'In-LineFor ced Torsional Vibration with Damping'." &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0,
"MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.ildata3.Show: MBTV.ildata3.SSTab1 = 0
Call MBTV.ildata3.openildata3(fname)
MBTV.ildata3.Caption = "MBTV - [" &Ftitle & "]"
MBTV.ildata3.saveproject.Enabled = True
MBTV.ildata3.SaveAs.Enabled = True
MBTV.ildata3.printproject.Enabled = True
MBTV.ildata3.preresults.Enabled = True
MBTV.ildata3.viewdata.Enabled = True
MBTV.ildata3.viewresult.Enabled = True
MBTV.ildata3.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “In-Line” and “Forced”, and put them to the FORM ildata4.
ElseIF In = "Analysis Type:In-LineFor ced Torsional Vibration without Damping" Then
IF dataForm.Name = "ildata4" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:"
&Chr(13) & "   'In-Line Forced Torsional Vibration without Damping'." &Chr(13) &Chr(13) & "Continue or
not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.ildata4.Show: MBTV.ildata4.SSTab1 = 0
Call MBTV.ildata4.openildata4(fname)
MBTV.ildata4.Caption = "MBTV - [" &Ftitle & "]"
MBTV.ildata4.saveproject.Enabled = True
```

```
MBTV.ildata4.SaveAs.Enabled = True
MBTV.ildata4.printproject.Enabled = True
MBTV.ildata4.preresults.Enabled = True
MBTV.ildata4.viewdata.Enabled = True
MBTV.ildata4.viewresult.Enabled = True
MBTV.ildata4.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Branch” and “Free”, and put them to the FORM mbdata1.
Else IF ln = "snalysis Type:Multi-Branch Free Torsional Vibration" Then
IF dataForm.Name = "mbdata1" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "    'Multi-BranchFree Torsional Vibration'." &Chr(13) &Chr(13) & "Continue or not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mbdata1.Show:MBTV.mbdata1.SSTab1 = 0
Call MBTV.mbdata1.openmbdata1(fname)
MBTV.mbdata1.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mbdata1.saveproject.Enabled = True
MBTV.mbdata1.SaveAs.Enabled = True
MBTV.mbdata1.printproject.Enabled = True
MBTV.mbdata1.preresults.Enabled = True
MBTV.mbdata1.viewdata.Enabled = True
MBTV.mbdata1.viewresult.Enabled = True
MBTV.mbdata1.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Branch” and “Free with Damping”, and put them to the FORM mbdata2.
Else IF ln = "snalysis Type:Multi-Branch Torsional Vibration with Damping" Then
IF dataForm.Name = "mbdata2" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "      'Multi-Branch Torsional Vibration with Damping'." &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mbdata2.Show:MBTV.mbdata2.SSTab1 = 0
Call mbdata2.openmbdata2(fname)
MBTV.mbdata2.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mbdata2.saveproject.Enabled = True
MBTV.mbdata2.SaveAs.Enabled = True
MBTV.mbdata2.printproject.Enabled = True
MBTV.mbdata2.preresults.Enabled = True
MBTV.mbdata2.viewdata.Enabled = True
```

```
MBTV.mbdata2.viewresult.Enabled = True
MBTV.mbdata2.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Branch” and “Forced with Damping”, and put them to the FORM mbdata3.
Else IF ln = "snalysis Type:Multi-BranchFor ced Torsional Vibration with Damping" Then
IF dataForm.Name = "mbdata3" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "   'Multi-Branch Forced Torsional Vibration with Damping'." &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mbdata3.Show:MBTV.mbdata3.SSTab1 = 0
Call mbdata3.openmbdata3(fname)
MBTV.mbdata3.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mbdata3.saveproject.Enabled = True
MBTV.mbdata3.SaveAs.Enabled = True
MBTV.mbdata3.printproject.Enabled = True
MBTV.mbdata3.preresults.Enabled = True
MBTV.mbdata3.viewdata.Enabled = True
MBTV.mbdata3.viewresult.Enabled = True
MBTV.mbdata3.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Branch” and “Forced”, and put them to the FORM mbdata4.
Else IF ln = "snalysis Type:Multi-Branch Forced Torsional Vibration without Damping" Then
IF dataForm.Name = "mbdata4" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "   'Multi-Branch Forced Torsional Vibration without Damping'." &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mbdata4.Show:MBTV.mbdata4.SSTab1 = 0
Call mbdata4.openmbdata4(fname)
MBTV.mbdata4.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mbdata4.saveproject.Enabled = True
MBTV.mbdata4.SaveAs.Enabled = True
MBTV.mbdata4.printproject.Enabled = True
MBTV.mbdata4.preresults.Enabled = True
MBTV.mbdata4.viewdata.Enabled = True
MBTV.mbdata4.viewresult.Enabled = True
MBTV.mbdata4.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
```

```
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Junction” and “Free”, and put them to the FORM mjdata1.
Else IF ln = "snalysis Type:Multi-JunctionFree Torsional Vibration" Then
IF dataForm.Name = "mjdata1" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:"
&Chr(13) & "    'Multi-JunctionFree Torsional Vibration'." &Chr(13) &Chr(13) & "Continue or not?", 4 + 32 + 0,
"MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mjdata1.Show:MBTV.mjdata1.SSTab1 = 0
Call MBTV.mjdata1.openmjdata1(fname)
MBTV.mjdata1.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mjdata1.saveproject.Enabled = True
MBTV.mjdata1.SaveAs.Enabled = True
MBTV.mjdata1.printproject.Enabled = True
MBTV.mjdata1.preresults.Enabled = True
MBTV.mjdata1.viewdata.Enabled = True
MBTV.mjdata1.viewresult.Enabled = True
MBTV.mjdata1.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Junction” and “Free with Damping”, and put them to the FORM mjdata2.
Else IF ln = "snalysis Type:Multi-Junction Torsional Vibration with Damping" Then
IF dataForm.Name = "mjdata2" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:"
&Chr(13) & "    'Multi-Junction Torsional Vibration with Damping'." &Chr(13) &Chr(13) & "Continue or not?", 4
+ 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mjdata2.Show:MBTV.mjdata2.SSTab1 = 0
Call mjdata2.openmjdata2(fname)
MBTV.mjdata2.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mjdata2.saveproject.Enabled = True
MBTV.mjdata2.SaveAs.Enabled = True
MBTV.mjdata2.printproject.Enabled = True
MBTV.mjdata2.preresults.Enabled = True
MBTV.mjdata2.viewdata.Enabled = True
MBTV.mjdata2.viewresult.Enabled = True
MBTV.mjdata2.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
```

```
End IF
‘Read the data of “Multi-Junction” and “Forced with Damping”, and put them to the FORM mjdata3.
Else IF ln = "snalysis Type:Multi-Junction Forced Torsional Vibration with Damping" Then
IF dataForm.Name = "mjdata3" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "    'Multi-JunctionFor ced Torsional Vibration with Damping'." &Chr(13) &Chr(13) & "Continue or not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mjdata3.Show:MBTV.mjdata3.SSTab1 = 0
Call mjdata3.openmjdata3(fname)
MBTV.mjdata3.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mjdata3.saveproject.Enabled = True
MBTV.mjdata3.SaveAs.Enabled = True
MBTV.mjdata3.printproject.Enabled = True
MBTV.mjdata3.preresults.Enabled = True
MBTV.mjdata3.viewdata.Enabled = True
MBTV.mjdata3.viewresult.Enabled = True
MBTV.mjdata3.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
‘Read the data of “Multi-Junction” and “Forced”, and put them to the FORM mjdata4.
Else IF ln = "snalysis Type:Multi-Junction Forced Torsional Vibration without Damping" Then
IF dataForm.Name = "mjdata4" Then
smsg = 6
Else
smsg = MsgBox(clumsg &Chr(13) &Chr(13) & "The analysis type of the File which will be opened Is:" &Chr(13) & "    'Multi-Junction Forced Torsional Vibration without Damping'." &Chr(13) &Chr(13) & "Continueor not?", 4 + 32 + 0, "MBTV 1.0")
End IF
IFsmsg = 6 Then
lis.Clear
picb.Picture = LoadPicture()
dataForm.Hide
MBTV.mjdata4.Show:MBTV.mjdata4.SSTab1 = 0
Call mjdata4.openmjdata4(fname)
MBTV.mjdata4.Caption = "MBTV - [" &Ftitle & "]"
MBTV.mjdata4.saveproject.Enabled = True
MBTV.mjdata4.SaveAs.Enabled = True
MBTV.mjdata4.printproject.Enabled = True
MBTV.mjdata4.preresults.Enabled = True
MBTV.mjdata4.viewdata.Enabled = True
MBTV.mjdata4.viewresult.Enabled = True
MBTV.mjdata4.viewgraphic.Enabled = True
Else
IF TextOpen = 0 Then
OpenEr = 1:Exit Sub
Else
Exit Sub
End IF
End IF
Else
smsg = MsgBox("The project File Is Invalid ! please open the another.", 0 + 48 + 256, "MBTV 1.0")
GoToopentype
```

```
End IF
OpenEr = 0:Exit Sub
OpenError:openEr = 1:Exit Sub
End Sub
```

1.23 Sub Revise( )

‘Revise the deflections on the end of each branch shafts to meet the need at junction point of the free torsional vibration.

```
Public Sub Revise(omg As Double, thetas11 As Double, sigmas11 As Double, thetae()As Double, sigmae() As Double)
Dim thetas(1 To ROW) As Double, theta(1 To ROW) As Double
Dim sigmas(1 To ROW) As Double
Dim I AsInteger, p AsInteger, q Asboolean, I1 AsInteger, sgj AsInteger
do
IF n <= 2 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetae(2) -thetae(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetae(i) -thetae(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetas(p) =thetae(1) *thetas11 /thetae(p)
Call Holzer(omg, thetas(p), sigmas11, thetae(p), sigmae(p), theta(), p)
Next p
End IF
Loop While q =False
End Sub
```

1.24 Sub ReviseFD( )

‘Revise the deflections on the end of each branch shafts to meet the need at junction point of the free torsional vibration with damping.

```
Public Sub ReviseFD(omg As Double, thetas11r As Double, thetas11i As Double, sigmas11r As Double, sigmas11i As Double, thetaer() As Double, thetaei() As Double, sigmaer() As Double, sigmaei() As Double)
Dim thetasr(1 To ROW) As Double, thetasi(1 To ROW) As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim I AsInteger, p AsInteger, q Asboolean
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaer(2) -thetaer(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetaer(i) -thetaer(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
```

```
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasr(p) =thetaer(1) *thetas11r /thetaer(p)
Call HolzerFD(omg, thetasr(p), thetasi(p), sigmas11r, sigmas11i, thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaei(2) -thetaei(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetaei(i) -thetaei(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasi(p) =thetaei(1) *thetas11i /thetaei(p)
Call HolzerFD(omg, thetasr(p), thetasi(p), sigmas11r, sigmas11i, thetaer(p), thetaei(p), sigmaer(p), sigmaei(p), thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
End Sub
```

1.25 Sub ReviseFT( )

```
'Revise the deflections on the end of each branch shafts to meet the need at junction point of the forced torsional vibration.
Public Sub ReviseFT(omg As Double, thetas11r As Double, thetas11i As Double, sigmasr() As Double, sigmasi() As Double, thetaer() As Double, thetaei() As Double, sigmaer() As Double, sigmaei() As Double)
Dim thetasr(1 To ROW) As Double, thetasi(1 To ROW) As Double
Dim ssigr(1 To ROW, 1 To ROW) As Double, ssigi(1 To ROW, 1 To ROW) As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim I AsInteger, p AsInteger, q Asboolean
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaer(2) -thetaer(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetaer(i) -thetaer(i - 1)) > 0.001 Then
```

```
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasr(p) =thetaer(1) *thetas11r /thetaer(p)
Call HolzerFT(omg, thetasr(p), thetasi(p), sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p),
thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaei(2) -thetaei(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetaei(i) -thetaei(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasi(p) =thetaei(1) *thetas11i /thetaei(p)
Call HolzerFT(omg, thetasr(p), thetasi(p), sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p),
thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
End Sub
```

1.26 Sub ReviseFTD( )

‘Revise the deflections on the end of each branch shafts to meet the need at junction point of the forced torsional vibration with damping.

```
Public Sub ReviseFTD(omg As Double, thetas11r As Double, thetas11i As Double, sigmasr() As Double,
sigmasi() As Double, thetaer() As Double, thetaei() As Double, sigmaer() As Double, sigmaei() As Double)
Dim thetasr(1 To ROW) As Double, thetasi(1 To ROW) As Double
Dim ssigr(1 To ROW, 1 To ROW) As Double, ssigi(1 To ROW, 1 To ROW) As Double
Dim thetar(1 To ROW) As Double, thetai(1 To ROW) As Double
Dim sigmar(1 To ROW) As Double, sigmai(1 To ROW) As Double
Dim I AsInteger, p AsInteger, q Asboolean
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaer(2) -thetaer(1)) > 0.001 Then
q =False
Else
q = True
End IF
```

```
Else
For I = 2 To n - 1
IF Abs(thetaer(i) -thetaer(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasr(p) =thetaer(1) *thetas11r /thetaer(p)
Call HolzerFTD(omg, thetasr(p), thetasi(p), sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p),
thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
do
IF n < 3 Then
Exit Do
ElseIF n = 3 Then
IF Abs(thetaei(2) -thetaei(1)) > 0.001 Then
q =False
Else
q = True
End IF
Else
For I = 2 To n - 1
IF Abs(thetaei(i) -thetaei(i - 1)) > 0.001 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = 1 To n - 1
thetasi(p) =thetaei(1) *thetas11i /thetaei(p)
Call HolzerFTD(omg, thetasr(p), thetasi(p), sigmasr(), sigmasi(), thetaer(p), thetaei(p), sigmaer(p), sigmaei(p),
thetar(), thetai(), sigmar(), sigmai(), p)
Next p
End IF
Loop While q =False
End Sub
```

1.27 Sub ReviseJ( )

```
'Revise the deflections on the end of each branch shafts to meet the need at junction point of the free
torsional vibration
Public Sub ReviseJ(omg As Double, thetasj As Double, sigmasj As Double, thetae()As Double, sigmae() As
Double, j1 AsInteger, sgj AsInteger)
Dim thetas(1 To ROW) As Double, theta(1 To ROW) As Double
Dim sigmas(1 To ROW) As Double
Dim I AsInteger, p AsInteger, q Asboolean, I1 AsInteger
do
IF nsj(j1) <= 2 Then
Exit Do
ElseIF nsj(j1) = 3 Then
IF Abs(thetae(sgj + 1) -thetae(sgj)) > 0.1 Then
q =False
Else
q = True
```

```
End IF
Else
For I = 1 To nsj(j1) - 2
IF Abs(thetae(sgj +I) -thetae(sgj +I - 1)) > 0.1 Then
q =False:ExitFor
Else
q = True
End IF
Next I
End IF
IF q = True Then
Exit Do
Else
For p = sgj To sgj + nsj(j1) - 2
thetas(p) =thetae(sgj) *thetasj /thetae(p)
Call Holzer(omg, thetas(p), sigmasj, thetae(p), sigmae(p), theta(), p)
Next p
End IF
Loop While q =False
End Sub
```

1.28 Sub Root( )

```
'Obtain the natural frequencies of a free vibration system.
Public Sub Root(thetas As Double, sigmas As Double, x1 As Double, x2 As Double, xx As Double, yy As Double, zz As Double)
Dim y1 As Double, y2 As Double, z1 As Double, z2 As Double
do
Call CalEnd (x1, thetas, sigmas, z1, y1)
Call CalEnd (x2, thetas, sigmas, z2, y2)
xx = (x1 * y2 - x2 * y1) / (y2 - y1)
Call CalEnd (xx, thetas, sigmas, zz, yy)
IF yy * y1 > 0 Then
y1 = yy: x1 = xx
Else
x2 = xx
End IF
Loop While Abs(yy) > 100
End Sub
```

1.29 Sub RootFD( )

```
'Obtain the natural frequencies of a free vibration system with damping.
Public Sub RootFD(thetasr As Double, thetasi As Double, sigmasr As Double, sigmasi As Double, xx() As Double, jx AsInteger)
Dim x1 As Double, x2 As Double, x3 As Double
Dim y1 As Double, y2 As Double, y3 As Double
Dim dex As Double, b AsInteger, om1(1 To RO) As Double
Dim I AsInteger, I1 AsInteger
Dim xx1(1 To RO) As Double, yy1(1 To RO) As Double
Dim the1(1 To RO, 1 To ROW, 1 To ROW) As Double
Dim sig1(1 To RO, 1 To ROW, 1 To ROW) As Double
Dim ther1(1 To RO, 1 To ROW, 1 To ROW) As Double
Dim sigr1(1 To RO, 1 To ROW, 1 To ROW) As Double
Dim thei1(1 To RO, 1 To ROW, 1 To ROW) As Double
Dim sigi1(1 To RO, 1 To ROW, 1 To ROW) As Double
Call RunFD(om1(), the1(), sig1(), ther1(), thei1(), sigr1(), sigi1(), b, dex)
jx = 0
For I = 2 Tob * 10
I1 =I1 + 1
xx1(i1) = (om1(i - 1) +om1(i)) / 2
yy1(i1) =sig1(i, n, m(n)) -sig1(i - 1, n, m(n))
Next I
For I = 2 ToI1
IF yy1(i) > 0 And yy1(i - 1) < 0 Then
x1 = xx1(i): x2 = xx1(i - 1)
y1 = yy1(i): y2 = yy1(i - 1)
x3 = (x1 * y2 - x2 * y1) / (y2 - y1)
```

```
jx =jx + 1: xx(jx) = x3
End IF
Next I
End Sub
```
1.30 Sub run( )

‘Calculate the deflections and residual torques of each discs and shafts at each the frequencies of the free vibration.

```
Public Sub run()
Dim om(1 To RO) As Double, the(1 To RO) As Double, sig(1 To RO) As Double
Dim omgg(1 To ROW) As Double, thee(1 To ROW) As Double, sigg(1 To ROW) As Double
Dim I AsInteger, p AsInteger,smsg As String
Dim delta As Double, omgmin As Double, omgmax As Double
js = 1: qs = 1
omgmin = 0:omgmax = 5000: delta =omgmax / 200
om(1) =omgmin:Call CalEnd (om(1), 1, 0, the(1), sig(1))
For I = 2 To 201
om(i) =omgmin + (i - 1) * delta
Call CalEnd (om(i), 1, 0, the(i), sig(i))
IFsig(i) *sig(i - 1) < 0 Then
omgg(js + 1) =om(i):omgg(js) =om(i - 1)
Thee(js + 1) = the(i): thee(js) = the(i - 1)
sigg(js + 1) =sig(i):sigg(js) =sig(i - 1)
js =js + 2
End IF
Next I
IF (js - 1) / 2 <= 0 Then
TextRun = 0
smsg = MsgBox("DataError! The softwareCan not run, It willbe terminated!", 0 + 16 + 256, "MBTV 1.0")
Exit Sub
End IF
For I = 1 Tojs - 1 Step 2
Call Root(1, 0, omgg(i), omgg(i + 1), omg3(qs), sig3(qs), the3(qs))
qs = qs + 1
Next I
End Sub
```
1.31 Sub RunFD( )

‘Calculate the deflections and residual torques of each discs and shafts at each frequencies of the free vibration with damping.

```
Public Sub RunFD(om() As Double, the() As Double, sig() As Double, ther()As Double, thei() As Double, sigr() As Double, sigi() As Double, b AsInteger, dex As Double)
Dim thetaer As Double, sigmaer As Double
Dim thetaei As Double, sigmaei As Double
Dim I AsInteger, I1 AsInteger, I2 AsInteger, jj AsInteger
Dim delta As Double, omgmin As Double, omgmax As Double
IF (js - 1) <= 0 Then TextRun = 0:Exit Sub
jj = (js - 1) / 2
IF omg3(jj) > 2500 Then
b = (Int(omg3(jj) / 500) + 1): dex = 500
ElseIF omg3(jj) > 2000 Then
b =Int(omg3(jj) / 400) + 1: dex = 400
ElseIF omg3(jj) > 1500 Then
b =Int(omg3(jj) / 300) + 1: dex = 300
ElseIF omg3(jj) > 1000 Then
b =Int(omg3(jj) / 200) + 1: dex = 200
ElseIF omg3(jj) > 500 Then
b =Int(omg3(jj) / 100) + 1: dex = 100
ElseIF omg3(jj) > 100 Then
b =Int(omg3(jj) / 50) + 1: dex = 50
Else
b =Int(omg3(jj) / 20) + 1: dex = 20
End IF
IF Fr <= 0 Then
omgmin = 0:omgmax =b * dex: delta =omgmax / (b * 10)
om(1) =omgmin
```

```
Call CalEnd fd(om(1), 1, 0, 0, 0, thetaer, thetaei, sigmaer, sigmaei)
The(1, n, m(n)) = Sqr(thetaer ^ 2 +thetaei ^ 2)
sig(1, n, m(n)) = Sqr(sigmaer ^ 2 +sigmaei ^ 2)
Ther(1, n, m(n)) = Abs(thetaer)
sigr(1, n, m(n)) = Abs(sigmaer)
Thei(1, n, m(n)) = Abs(thetaei)
sigi(1, n, m(n)) = Abs(sigmaei)
Else
omgmin = (fr - 1) * dex:omgmax =Fr * dex
delta = dex / (b * 10): dex = dex /b
om(1) =omgmin
Call CalEnd fd(om(1), 1, 0, 0, 0, thetaer, thetaei, sigmaer, sigmaei)
The(1, n, m(n)) = Sqr(thetaer ^ 2 +thetaei ^ 2)
sig(1, n, m(n)) = Sqr(sigmaer ^ 2 +sigmaei ^ 2)
Ther(1, n, m(n)) = Abs(thetaer)
sigr(1, n, m(n)) = Abs(sigmaer)
Thei(1, n, m(n)) = Abs(thetaei)
sigi(1, n, m(n)) = Abs(sigmaei)
End IF
For I = 2 Tob * 10
om(i) =om(1) + (i - 1) * delta
Call CalEnd fd(om(i), 1, 0, 0, 0, thetaer, thetaei, sigmaer, sigmaei)
The(i, n, m(n)) = Sqr(thetaer ^ 2 +thetaei ^ 2)
sig(i, n, m(n)) = Sqr(sigmaer ^ 2 +sigmaei ^ 2)
Ther(i, n, m(n)) = Abs(thetaer)
sigr(i, n, m(n)) = Abs(sigmaer)
Thei(i, n, m(n)) = Abs(thetaei)
sigi(i, n, m(n)) = Abs(sigmaei)
Next I
End Sub
```

1.32 Sub RunFT( )

‘Calculate the deflections and residual torques of each discs and shafts at each frequencies of the forced vibration.

```
Public Sub RunFT(omg As Double, ther() As Double, thei() As Double, sigr()As Double, sigi() As Double)
Dim thetaEnd r As Double, thetaEnd i As Double
Dim sigmaEnd r As Double, sigmaEnd i As Double, I AsInteger
For I = 1 To m(1)
Ftr(1, I) =Ft(1, I) *Cos(afa(1, I))
Fti(1, I) =Ft(1, I) *sin(afa(1, I))
Next I
Call CalEndFT(omg,Ftr(),Fti(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, ther(), thei(), sigr(), sigi())
End Sub
```

1.33 Sub RunFTD( )

‘Calculate the deflections and residual torques of each discs and shafts at each frequencies of the forced vibration with damping.

```
Public Sub RunFTD(omg As Double, ther() As Double, thei() As Double, sigr()As Double, sigi() As Double)
Dim thetaEnd r As Double, thetaEnd i As Double
Dim sigmaEnd r As Double, sigmaEnd i As Double, I AsInteger
Dim p AsInteger, q AsInteger
IF n < 3 Then
For I = 1 To m(1)
Ftr(1, I) =Ft(1, I) *Cos(afa(1, I))
Fti(1, I) =Ft(1, I) *sin(afa(1, I))
Next I
Else
For p = 1 To n
For q = 1 To m(p)
Ftr(p, q) =Ft(p, q) *Cos(afa(p, q))
Fti(p, q) =Ft(p, q) *sin(afa(p, q))
Next q
Next p
End IF
Call CalEndFTD(omg,Ftr(),Fti(), thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, ther(), thei(), sigr(), sigi())
End Sub
```

Part 2 modPrinting module

In this module, there are the programs for printing, printing view.

variables

```
Private Type Rect
Left As Long
Top As Long
Right As Long
bottom As Long
End Type
Private TypeCharRange
CpMin As Long
CpMax As Long
End Type
Private TypeFormatRange
hdc As Long
hdcTarget As Long
Rc As Rect
RcPage As Rect
Chrg AsCharRange
End Type
PrivateConst WM_USER As Long = &H400
PrivateConstEM_FORMATRANGE As Long = WM_USER + 57
PrivateConstEM_SETTARGETDEVICE As Long = WM_USER + 72
PrivateConst PHYSICALOFFSETX As Long = 112
PrivateConst PHYSICALOFFSETY As Long = 113
Private DeclareFunction GetDeviceCaps Lib "gdi32" (ByVal hdc As Long, byVal nIndex As Long) As Long
Public DeclareFunction SEnd Message Lib "USER32" Alias "SEnd MessageA" (ByVal hWnd As Long, byVal msg As Long, byVal wp As Long, lp As Any) As Long
PublicConstCB_FINDSTRINGEXACT = &H158
PublicConstCB_FINDSTRING = &H14C
PublicConstCB_ERR = (-1)
Private DeclareFunctionCreateDC Lib "gdi32" Alias "CreateDCA" (ByVal lpDriverName As String, byVal lpDeviceName As String, byVal lpOutput As Long, byVal lpInitData As Long) As Long
```

2.1 Sub PrintPreview( )

```
'This subroute is for the printing view.
Public Sub PrintPreview(RTF As RichTextBox, LeftMarginWidth AsCurrency, TopMarginHeight AsCurrency, RightMarginWidth AsCurrency, bottomMarginHeightAsCurrency, pgOrientation AsInteger)
Dim LeftOffset As Long, TopOffset As Long
Dim LeftMargin As Long, TopMargin As Long
Dim RightMargin As Long, bottomMargin As Long
Dim Fr AsFormatRange
Dim rcDrawTo As Rect
Dim rcPage As Rect
Dim TextLength As Long
Dim Next CharPosition As Long
Dim r As Long
Dim ICount AsInteger
onError GoToErrHandle
printer.Orientation = pgOrientation
printer.ScaleMode = vbTwips
LeftMargin =CLng(LeftMarginWidth - LeftOffset)
TopMargin =CLng(TopMarginHeight - TopOffset)
RightMargin =CLng((Printer.Width - RightMarginWidth) - LeftOffset)
bottomMargin =CLng((Printer.Height -bottomMarginHeight) - TopOffset)
RcPage.Top = 0
RcPage.Right = Printer.ScaleWidth
RcPage.Bottom = Printer.ScaleHeight
RcDrawTo.Left = LeftMargin
RcDrawTo.Top = TopMargin
RcDrawTo.Right = RightMargin
RcDrawTo.Bottom =bottomMargin
FrmPreview.SizePreview Printer.Width, Printer.Height
Fr.hdc =FrmPreview.picPreview(0).hdc
```

```
Fr.hdcTarget =FrmPreview.picPreview(0).hdc
Fr.rc = rcDrawTo
Fr.rcPage = rcPage
Fr.chrg.cpMin = 0
Fr.chrg.cpMax = -1
TextLength = Len(RTF.Text)
Dim lPage AsInteger
lPage = 1
do
WithFrmPreview
IF lPage > 1 Then
.AddPagelPage
Fr.hdc = .picPreview(iPage - 1).hdc
Fr.hdcTarget = .picPreview(iPage - 1).hdc
End IF
.picPreview(iPage - 1).Print
End With
Next CharPosition = SEnd Message(RTF.hWnd, EM_FORMATRANGE, True, Fr)
IFNext CharPosition >= TextLength ThenExit Do    'IF done thenExit
Fr.chrg.cpMin =Next CharPosition ' Starting positionFor Next page
lPage =lPage + 1
Loop
R = SEnd Message(RTF.hWnd, EM_FORMATRANGE, False, byValCLng(0))
FrmPreview.Show
Exit Sub
ErrHandle: SelectCaseErr.Number
Case 482
MsgBox "Make sure that you have a printerInstalled.IF a printerIsInstalled, goInto your printer properties look under the Setup tab, and make sure theICMCheckboxIsChecked and try printing again.", , "PrinterError"
Exit Sub
CaseElse
MsgBoxErr.Number & " " &Err.Description
ResumeNext
End Select
End Sub
2.2 Sub printgraphics( )
'This subroute is for printing the graphics.
Public Sub printgraphics(daFor m AspictureBox)
Dim lCount, lPicCount AsInteger
onError GoToErrHandle
daForm.Picture = daForm.Image
IF printer.Copies > 0 Then
For lCount = 1 To Printer.Copies
printer.Print
printer.FontSize = 12
printer.PaintPicture daForm.Image, 300, 400
printer.End Doc
Next
End IF
Exit Sub
ErrHandle: SelectCaseErr.Number
Case 482
MsgBox "Make sure that you have a printerInstalled.IF a printerIsInstalled, goInto your printer properties. PrinterError"
Exit Sub
Case 32755
Exit Sub
CaseElse
MsgBoxErr.Number & " " &Err.Description, , "Preview - Printing"
ResumeNext
End Select
End Sub
```

Part 3 Forms

In this part, there are many For m main programs which are sharedbyEachForm.by them realize allInterfaces.

3.1 frmAbout

This program display the software aboutInFormation

3.1.1 Sub Form_Load( )

```
Public Sub Form_Load()
MBTV.frmAbout.Left = (Screen.Width -MBTV.frmAbout.Width) / 2
MBTV.frmAbout.Top = (Screen.Height -MBTV.frmAbout.Height) / 2
End Sub
```

3.2 frmFace

This program realize the welcome Interface

3.2.1 Sub about_Click( )

```
Private Sub about_Click()
FrmAbout.Show
End Sub
```

3.2.2 Sub Contents_Click( )

```
Private Sub Contents_Click()
Dim aa As Long, bb As String
sa = htmlhelp(Me.hWnd, App.HelpFile + "::\WelcomeMBTV.htm", 0, 0)
IF aa = 0 Then
bb = MsgBox("Can'topen the helpFile", vbOKOnly)
End IF
End Sub
```

3.2.3 Sub Form_Load( )

```
Private Sub Form_Load( )
MBTV.frmFace.Left = (Screen.Width -MBTV.select.Width) / 2
MBTV.frmFace.Top = (Screen.Height -MBTV.select.Height) / 2
RunFlag = 0
```

3.2.4 Sub Exit_Click( )

```
Private Sub Exit_Click()
End
End Sub
```

3.2.5 Sub new_Click( )

```
Private Sub new_Click()
MBTV.frmFace.Hide
MBTV.select.Show
End Sub
```

3.2.6 Sub open_Click( )

```
Private Sub open_Click()
Call ProjectOpen(Me, Me.List1, Me.Picture1)
saveproject.Enabled = True: saveas.Enabled = True
printproject.Enabled = True: preresults.Enabled = True
End Sub
```

3.3 frmPreview

This program build preview Interface and realize Its Functions.

```
OptionExplicit
PrivateConst lBorder = 100
Private ScalePercent AsInteger
PrivatebLoad Asboolean
```

3.3.1 Sub AddPage( )

```
Public Sub AddPage(PageNumber AsInteger)
IF pageNumber > 1 Then
Load picPreview(PageNumber - 1)
set picPreview(PageNumber - 1) = Nothing
TabPreview.Tabs.Add PageNumber, , "Page " & PageNumber
End IF
End Sub
```

3.3.2 Sub FillCboPercent( )

```
Private Sub FillCboPercent()
Dim ICount AsInteger
Dim strSearch As String
WithCboPercent
For ICount = 200 To 30 Step -10
.AddItemCStr(iCount) & "%"
```

```
Next
strSearch = "100%"
.ListIndex = SEnd Message(.hWnd, CB_FINDSTRING, -1, byVal strSearch)
End With
End Sub
```

3.3.3 Sub pictureShow( )

```
Public Sub pictureShow()
screen.MousePointer = vbHourglass
With picChild
.Height = (ScalePercent / 100) * picPreview(0).Height
.Width = (ScalePercent / 100) * picPreview(0).Width
ResizeScrollBars
End With
screen.MousePointer = vbDefault
End Sub
```

3.3.4 Sub PreviewPrint( )

```
Private Sub PreviewPrint()
Dim ICount, IPicCount AsInteger
onError GoToErrHandle
For ICount = 0 To picPreview.Count - 1
picPreview(iCount).Picture = picPreview(iCount).Image
Next
IF printer.Copies > 0 Then
For ICount = 1 To Printer.Copies
printer.Print
For IPicCount = 0 To picPreview.Count - 1
printer.PaintPicture picPreview(iPicCount).Picture, 0, 0
IF IPicCount < picPreview.Count - 1 Then Printer.NewPage
Next
printer.End Doc
Next
End IF
Exit Sub
ErrHandle:
selectCaseErr.Number
Case 482
MsgBox "Make sure that you have a printerInstalled.IF a printerIsInstalled, goInto your printer propertiesPrinterError"
Exit Sub
Case 32755
Exit Sub
CaseElse
MsgBoxErr.Number & " " &Err.Description, , "Preview - Printing"
 ResumeNext
End Select
End Sub
```

3.3.5 Sub PreviewZoomIn( )

```
Private Sub PreviewZoomIn()
WithCboPercent
IF.ListIndex - 1 >= 0 Then
scalePercent = ScalePercent + 10
.ListIndex =.ListIndex - 1
End IF
End With
Exit Sub
ErrHandle: SelectCaseErr.Number
CaseElse
MsgBoxErr.Number & " " &Err.Description, , "Preview - Printing"
ResumeNext
End Select
End Sub
```

3.3.6 Sub PreviewZoomOut( )

```
Private Sub PreviewZoomOut()
WithCboPercent
```

```
IF.ListIndex + 1 <.ListCount Then
scalePercent = ScalePercent - 10
.ListIndex =.ListIndex + 1
End IF
End With
Exit Sub
ErrHandle: SelectCaseErr.Number
CaseElse
MsgBoxErr.Number & " " &Err.Description, , "Preview - Printing"
ResumeNext
End Select
End Sub
```

3.3.7 Sub ResizeScrollBars()

```
Private Sub ResizeScrollBars()
With vscPreview
IF picChild.Height > picParent.Height Then
.Visible = True
.Max = picChild.Height - picParent.ScaleHeight
.Min = 0
.LargeChange = picChild.Height - picParent.Height
ImgCorner.Visible = True
Else
.Visible =False
ImgCorner.Visible =False
End IF
End With
WithhscPreview
IF picChild.Width > picParent.Width Then
.Visible = True
.Max = picChild.Width - picParent.ScaleWidth
.Min = 0
.LargeChange = picChild.Width - picParent.ScaleWidth
ImgCorner.Visible = True
Else
.Visible =False
ImgCorner.Visible =False
End IF
End With
End Sub
```

3.3.8 Sub SizePreview( )

```
Public Sub SizePreview(lWidth As Long, lHeight As Long)
Dim ICount AsInteger
For ICount = 0 To picPreview.Count - 1
With picPreview(iCount)
.Left = 0:.Top = 0:.Width = lWidth:.Height = lHeight
End With
Next
picChild.Move 0, 0, lWidth, lHeight
End Sub
```

3.3.9 Sub btnPreview_Click( )

```
Public Sub btnPreview_Click(index AsInteger)
selectCaseIndex
Case 0    'Print
previewPrint
Unload Me
Case 1    'ZoomIn
previewZoomIn
Case 2    'Zoomout
previewZoomOut
End Select
End Sub
```

3.3.10 Sub CboPercent_Change( )

```
Private Sub CboPercent_Change()
IF bLoad =False Then
```

```
WithCboPercent
scalePercent =CInt(Left(.List(.ListIndex), Len(.List(.ListIndex)) - 1))
End With
pictureShow
End IF
End Sub
```

3.3.11 Sub CboPercent_Click( )

```
Private Sub CboPercent_Click()
IF bLoad =False Then
WithCboPercent
scalePercent =CInt(Left(.List(.ListIndex), Len(.List(.ListIndex)) - 1))
End With
pictureShow
End IF
End Sub
```

3.3.12 Sub CmdClose_Click( )

```
Private Sub CmdClose_Click()
Unload Me
CallFrmPrevRTB.Command1_Click
End Sub
```

3.3.13 Sub Form_Activate( )

```
Private Sub Form_Activate()
With picPreview(0)
.Picture = .Image
picChild.Move 0, 0, .Width, .Height
picChild.Picture = .Picture
pictureShow
End With
End Sub
```

3.3.14 Sub Form_Load( )

```
Private Sub Form_Load()
bLoad = True
FillCboPercent
scalePercent = 100
WindowState = vbMaximized
bLoad =False
End Sub
```

3.3.15 Sub Form_Resize( )

```
Private Sub Form_Resize()
onError ResumeNext
IF WindowState = vbMinimized ThenExit Sub
picToolbar.Move 0, 0, Width
With tabPreview
.Move lBorder, ScaleHeight - .Height - lBorder, ScaleWidth - (2 * lBorder)
picParent.Move lBorder, lBorder + picToolbar.Height, ScaleWidth - (2 * lBorder), ScaleHeight - .Height -
picToolbar.Height - (2 * lBorder)
End With
End Sub
```

3.3.16 Sub hscPreview_Change( )

```
Private Sub hscPreview_Change()
picChild.Left = (-hscPreview.Value)
End Sub
```

3.3.17 Sub hscPreview_Scroll( )

```
Private Sub hscPreview_Scroll( )
picChild.Left = (-hscPreview.Value)
End Sub
```

3.3.18 Sub picParent_Resize( )

```
Private Sub picParent_Resize( )
Dim lCount AsInteger
With picParent
vscPreview.Move .ScaleLeft + .ScaleWidth - vscPreview.Width, .ScaleTop, vscPreview.Width, .ScaleHeight
-hscPreview.Height
hscPreview.Move 0, .ScaleHeight -hscPreview.Height, .ScaleWidth - vscPreview.Width
ImgCorner.Move vscPreview.Left,hscPreview.Top
```

```
End With
ResizeScrollBars
End Sub
```
3.3.19 Sub tabPreview_Click( )
```
Private Sub tabPreview_Click( )
With picPreview(tabPreview.SelectedItem.index - 1)
.Picture = .Image
picChild.Picture = .Picture
pictureShow
End With
End Sub
```
3.3.20 Sub vscPreview_Change( )
```
Private Sub vscPreview_Change( )
picChild.Top = (-vscPreview.Value)
End Sub
```
3.3.21 Sub vscPreview_Scroll()
```
Private Sub vscPreview_Scroll()
picChild.Top = (-vscPreview.Value)
End Sub
```
3.4 ildata1

This FORM is used to build the interfaces of In-Line Free vibration.
```
OptionExplicit
Dim oldx Assingle, oldy Assingle
private s AsInteger, t AsInteger
Private TextChang AsInteger, TextSave AsInteger
PrivateConst ROW AsInteger = 40, RO AsInteger = 250
Privatetheta(1 To ROW, 1 To RO) As Double, thetae(1 To ROW) As Double
```
3.4.1 Sub Combdex( )
```
Private Sub Combdex( )
Dim I, j AsInteger
Dim dex As Double, b AsInteger, jj AsInteger
N = 1: nj = 0:Fr = 0
jj = (js - 1) / 2
IF jj <= 0 ThenExit Sub
IF omg3(jj) > 2500 Then
b = (Int(omg3(jj) / 500) + 1): dex = 500
ElseIF omg3(jj) > 2000 Then
b =Int(omg3(jj) / 400) + 1: dex = 400
ElseIF omg3(jj) > 1500 Then
b =Int(omg3(jj) / 300) + 1: dex = 300
ElseIF omg3(jj) > 1000 Then
b =Int(omg3(jj) / 200) + 1: dex = 200
ElseIF omg3(jj) > 500 Then
b =Int(omg3(jj) / 100) + 1: dex = 100
ElseIF omg3(jj) > 100 Then
b =Int(omg3(jj) / 50) + 1: dex = 50
Else
b =Int(omg3(jj) / 20) + 1: dex = 20
End IF
Combo1.Clear
Combo1.Text = "ω:0--" & Str(b * dex)
Combo1.AddItem "ω:0--" & Str(b * dex)
For I = 1 Tob
Combo1.AddItem "ω:" & Str((i - 1) * dex) & "--" & Str(i * dex)
Next I
End Sub
```
3.4.2 Sub openildata1( )
```
Public Sub openildata1(fname As String)
Dim ln, sj1, sRunFlag As String, I, j AsInteger, k1 AsInteger
Dim somg3(1 To ROW) As String, stheta(1 To ROW, 1 To RO) As String
TextSave = 0
onError GoToopenError
IFFname = "" ThenExit Sub
openFnameFor Input As #1
```

```
For I = 1 To 6
LineInput #1, ln
Next I
LineInput #1, sProname: Text4.Text = sProname
LineInput #1, sProdate
LineInput #1, sRunFlag:RunFlag = Val(sRunFlag)
LineInput #1, ln
LineInput #1,sm(1)
For k1 = 1 To Val(sm(1)) - 1
LineInput #1,sinn(1, k1)
LineInput #1, sk(1, k1)
Next k1
LineInput #1,sinn(1, Val(sm(s)))
IFRunFlag = 0 Then GoTodataEnd
For I = 1 To 3
LineInput #1, ln
Next I
LineInput #1, sj1
js = (Val(sj1)) * 2 + 1
For I = 1 To Val(sj1)
LineInput #1, somg3(i)
omg3(i) = Val(somg3(i))
For k1 = 1 To Val(sm(1))
LineInput #1, stheta(i, k1)
theta(i, k1) = Val(stheta(i, k1))
Next k1
Next I
Close
DataEnd : SSTab1.Tab = 0
Label3.Visible = True: Text3.Visible = True
Label2.Caption = "Inertiaof Discs( 1 )"
Label3.Caption = " StIFFnessof Shafts( 1 )"
Text1.Text =sm(1): Text2.Text =sinn(1, 1): Text3.Text = sk(1, 1)
Frame2.Caption = "Disc 1"
N = 1
M(1) = Val(sm(1))
For I = 1 To Val(sm(1))
Inn(1, I) = Val(sInn(1, I)): k(1, I) = Val(sk(1, I))
Next I
s = 1: t = 1: u = 1:Fr = 0
IF m(1) > 1 Then
Command1.Enabled =False:Command2.Enabled = True
Else
Command1.Enabled =False:Command2.Enabled =False
End IF
Command5.Enabled = True
IFRunFlag = 0 Then
TextOk1 = 1: TextOk2 = 1
TextRun = 0:Command3.Enabled =False
Close:Exit Sub
Else
TextOk1 = 1: TextOk2 = 1
TextRun = 1:Command3.Enabled = True
End IF
Call results
Call Combdex
Call Command9_Click
OpenError:Exit Sub
End Sub
```
3.4.3 Sub about_Click( )
```
Private Sub about_Click()
FrmAbout.Show
End Sub
```
3.4.4 Sub Combo1_Click( )

```
Private Sub Combo1_Click( )
Dim I1 AsInteger, I2 AsInteger, I AsInteger,smsg As String
Dim dex As Double, b AsInteger
Dim jj AsInteger
jj = (js - 1) / 2
IF omg3(jj) > 2500 Then
b = (Int(omg3(jj) / 500) + 1): dex = 500
ElseIF omg3(jj) > 2000 Then
b =Int(omg3(jj) / 400) + 1: dex = 400
ElseIF omg3(jj) > 1500 Then
b =Int(omg3(jj) / 300) + 1: dex = 300
ElseIF omg3(jj) > 1000 Then
b =Int(omg3(jj) / 200) + 1: dex = 200
ElseIF omg3(jj) > 500 Then
b =Int(omg3(jj) / 100) + 1: dex = 100
ElseIF omg3(jj) > 100 Then
b =Int(omg3(jj) / 50) + 1: dex = 50
Else
b =Int(omg3(jj) / 20) + 1: dex = 20
End IF
IF Combo1.ListIndex = -1 Then
smsg = MsgBox("Please Select the ScopeofFrequencies!", 32, "MBTV 1.0")
combo1.Text = "ω:0--" & Str(b * dex)
Exit Sub
End IF
I1 =Combo1.ListIndex
Fr =I1
End Sub
```

3.4.5 Sub Command1_Click( )

```
Private Sub Command1_Click( )
Dim smsg As String, stext2 As String, stext3 As String
IF TextOk2 = 1 Then
command5.Enabled = True
End IF
IF TextOk1 = 0 Then
smsg = MsgBox("PressoK to make sure The Nubersof Discs!", 48, "MBTV 1.0")
Exit Sub
End IF
Text2.SetFocus
stext2 = Text2.Text
IF Val(stext2) <= 0 And stext2 <> "" Then
smsg = MsgBox("Inertiaof Discs(" & Str(t) & ")Invalid! "&Chr(13) & " PleaseInput a proper value!", 48, "MBTV
1.0")
Exit Sub
End IF
sInn(s, t) = stext2
stext3 = Text3.Text
IF Val(stext3) <= 0 And stext3 <> "" Then
smsg = MsgBox(" StIFFnessof Shafts(" & Str(t) & ")Invalid! " &Chr(13) & " PleaseInput a proper value!", 48,
"MBTV 1.0")
Exit Sub
End IF
sk(s, t) = stext3
IF Val(sm(s)) < 2or Val(sm(s)) <>Fix(Val(sm(s))) Then
smsg = MsgBox(" Numberof DiscsIsInvalid! PleaseInput a proper "& "value!", 48, "MBTV 1.0")
Exit Sub
End IF
IF t = Val(sm(s)) And Val(stext2) > 0 Then
Command5.Enabled = True: TextOk2 = 1
End IF
IF t = 1 Then
Command1.Enabled =False:Command2.Enabled = True
Else
Command1.Enabled = True:Command2.Enabled = True
```

```
T = t - 1
End IF
Frame2.Caption = "Disc " & Str(t)
IF t < Val(sm(s)) Then
Label3.Visible = True: Text3.Visible = True
End IF
IF t < Val(sm(s)) And TextOk2 = 0 ThenCommand5.Enabled =False
Label2.Caption = "Inertiaof Disc (" & Str(t) & ")"
Label3.Caption = " StIFFnessof Shaft (" & Str(t) & ")"
Text1.Text =sm(s)
Text2.Text =sinn(s, t)
Text3.Text = sk(s, t)
End Sub
3.4.6 Sub Command10_Click( )
Private Sub Command10_Click( )
Dim om(1 To RO) As Double, omgmax As Double, omgmin As Double, delta As Double
Dim the1(1 To RO) As Double, sig1(1 To RO) As Double
Dim I AsInteger, jj AsInteger, k1 Assingle, j1 AsInteger
Dim I1 AsInteger, I2 As Double, I3 As Double, I4 As Double
Dim a1 AsInteger, a2 AsInteger, b1 AsInteger, b2 AsInteger
Dim ymaxs As Double, ymins As Double, ymaxt As Double, ymint As Double
Dim dexs As Double, dext As Double, deys As Double, deyt As Double
Dim xt AsInteger, yt AsInteger, xx Assingle, yy Assingle
Dim u1 As Double, v1 As Double, xc As Double, x1 As Double
Dim units As Double, unitt As Double,smsg As String, xx1 As Double
Dim Curx As Double, Cury As Double
IF TextRun = 0 Then
smsg = MsgBox("Press Runbutton to start the analysis programeFirst!", 0 + 48 + 256, "MBTV 1.0")
Exit Sub
End IF
Call CurveF(a1, a2, deys, deyt, ymas, ymis, ymat, ymit, b1, b2, dexs, dext, om(), the1(), sig1(), units, unitt)
picture1.Picture = LoadPicture()
picture1.AutoRedraw = True
oldx = 1100:oldy = 750
scaleMode = 0
xx = 4000: yy = 2800
scale (0, 0)-(xx + 600, yy + 500)
picture1.FontSize = 10
picture1.CurrentX =oldx - 100:picture1.CurrentY =oldy - 750
picture1.Print "In-LineFree Torsional Vibration Analysis System"
picture1.CurrentX =oldx:picture1.CurrentY =oldy - 500
picture1.FontSize = 8
picture1.Print "Project Name:    " & sProname + "date:    " & sProdate
xt = xx /b2: yt = yy / a2
picture1.Line (oldx, oldy +I * yt) -(oldx + xx, oldy +I * yt), RGB(0, 0, 255)
For I = 0 To a2
xc = (Int(ymat / deyt) + 1) * deyt -I * deyt
picture1.Line (oldx, oldy +I * yt) -(oldx + xx, oldy +I * yt), RGB(0, 0, 255)
Call CurXY(xc, Curx, Cury)
Withpicture1
.CurrentX =oldx +Curx:.CurrentY =oldy +I * yt +Cury
End With
picture1.Print (Int(ymat / deyt) + 1) * deyt -I * deyt
IF Int(ymat / deyt) + 1 -I = 0 Then
picture1.Line (oldx, oldy +I * yt) -(oldx + xx, oldy +I * yt), RGB(255, 0, 0)
End IF
Next I
picture1.Line (oldx + xx, oldy + a2 * yt) -(oldx + xx + 400, oldy + a2 * yt), RGB(0, 0, 255)
picture1.Line (oldx + xx + 400, oldy + a2 * yt) -(oldx + xx + 250, oldy + a2 * yt - 30), RGB(0, 0, 255)
picture1.Line (oldx + xx + 400, oldy + a2 * yt) -(oldx + xx + 250, oldy + a2 * yt + 30), RGB(0, 0, 255)
Withpicture1
.CurrentX =oldx + xx + 350:.CurrentY =oldy + a2 * yt + 50
End With
picture1.FontSize = 10:Picture1.Print "":Picture1.FontSize = 8
```

```
For I = 0 Tob2
picture1.Line (oldx +I * xt, oldy) -(oldx +I * xt, oldy + yy), RGB(0, 0, 255)
Withpicture1
.CurrentX =oldx - 150 +I * xt:.CurrentY =oldy + yy + 50
End With
picture1.Print Str((i + 1) * dext)
Next I
picture1.Line (oldx, oldy)-(oldx, oldy - 300), RGB(0, 0, 255)
picture1.Line (oldx, oldy - 300)-(oldx - 30, oldy - 150), RGB(0, 0, 255)
picture1.Line (oldx, oldy - 300)-(oldx + 30, oldy - 150), RGB(0, 0, 255)
Withpicture1
.CurrentX =oldx - 180:.CurrentY =oldy - 300
End With
picture1.FontSize = 10:Picture1.Print "θ":Picture1.FontSize = 8
For I = 0 Tob2 * 5
picture1.Line (oldx +I * xt / 5, oldy + yy - 50) -(oldx +I * xt / 5, oldy + yy), RGB(0, 0, 255)
Next I
For I = 1 To a2 * 5
picture1.Line (oldx, oldy +I * yt / 5) -(oldx + 50, oldy +I * yt / 5), RGB(0, 0, 255)
Next I
k1 = (Int(ymat / deyt) + 1) * yt
u1 = xt: v1 = yt / deyt / unitt
For j1 = 1 To (js - 1) / 2
IF dext = 5 Then
For I = 2 Tob2 - 2 Step 2
picture1.Line (oldx + (i - 2) * u1, oldy + k1 -theta(j1, I / 2) * v1)-(oldx +I * u1, oldy + k1 -theta(j1, I / 2 + 1) * v1), RGB(0, 0, 0)
Next I
picture1.Line (oldx + (b2 - 2) * u1, oldy + k1 -theta(j1, b2 / 2) * v1)-(oldx +b2 * u1, oldy + k1 - 0), RGB(0, 0, 0)
Else
For I = 1 Tob2 - 1
picture1.Line (oldx - xx1 + (i - 1) * u1, oldy + k1 -theta(j1, I) * v1)-(oldx - xx1 +I * u1, oldy + k1 -theta(j1, I + 1) * v1), RGB(0, 0, 0)

Else
Command8.Enabled = True
End IF
Label5.Caption = "Inertiaof Disc (" & Str(s) & ", " & Str(t) & ")"
Label6.Caption = "External Dampingof Discs( " & Str(s) & ", " & Str(t) & ")"
Label7.Caption = "Internal Dampingof Shafts (" & Str(s) & ", "& Str(t) & ")"
Label8.Caption = " StIFFnessof Shaft (" & Str(s) & ", " &Str(t) & ")"
Text4.Text =sm(s)
Text5.Text =sinn(s, t)
Text6.Text = sedp(s, t)
Text7.Text = sidp(s, t)
Text8.Text = sk(s, t)
End Sub
```

3.13.13 Sub Command6_Click( )

```
Private Sub Command6_Click()
Dim smsg As String
Text4.SetFocus
IF TextOk1 = 0 Then
smsg = MsgBox("You should pressoK to make sure thebasicdatas"First!", 48, "MBTV 1.0")
Exit Sub
End IF
IF Val(Text4.Text) < 1or Val(Text4.Text) <>Fix(Val(Text4.Text)) Then
smsg = MsgBox(" Numberof Discs shouldbe a positiveInteger", 48, "MBTV 1.0")
Text4.SetFocus
Exit Sub
End IF
sm(s) = Text4.Text
IF Val(Text5.Text) < 0 Then
smsg = MsgBox("Inertiaof Discs( " & Str(s) & ", " & Str(t) & "IsInvalid!"&Chr(13) & " PleaseInput a proper value!", 48, "MBTV 1.0")
```

```
Exit Sub

For I3 = 1 To m(i2)
I4 =I4 + 1: stt1(i1, I4) = Sqr(ther(i2, I3) ^ 2 + thei(i2, I3) ^ 2)
Call Cpxalign(ther(i2, I3), thei(i2, I3), sign1, sign2, xx1, xx2)
Call Cpxalign(sigr(i2, I3), sigi(i2, I3), sign3, sign4, xx3, xx4)
List1.AddItem " " &Format(i2, "00") & "=" &sign1 &Format(ther(i2, I3), "0.000000") & "E" & xx1 &sign2 & "j"
&Format(Abs(thei(i2, I3)), "0.000000") & "E" & xx2 & "      " &Format(i2, "00") & "=" & sign3 &Format(sigr(i2,
I3), "0.000000") & "E" & xx3 &sign4 & "j" & Format(Abs(sigi(i2, I3)), "0.000000") & "E" & xx4
Next I3
End IF
Next I2
Call ReviseFD(omgg(i1), 1, 0, 0, 0, thetaer(), thetaei(), sigmaer(), sigmaei())
Thesr =thetaer(sgj + nsj(j) - 2): thesi =thetaei(sgj + nsj(j) - 2)
For I5 = 1 To sgj + nsj(j) - 2
sigsr =sigsr +sigmaer(i5)
sigsi =sigsi +sigmaei(i5)
Next I5
For I6 = 1 To m(sgj + nsj(j) - 1)
Call HolzerFD(omgg(i1), thesr, thesi, sigsr, sigsi, thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, ther(), thei(),
sigr(), sigi(), sgj + nsj(j) - 1)
Thesr =thetaEnd r:sigsr =sigmaEnd r: thesi =thetaEnd i
sigsi =sigmaEnd i: sgj = sgj + nsj(j)
I4 =I4 + 1
stt1(i1, I4) = Sqr(ther(sgj + nsj(j) - 1, I6) ^ 2 + thei(sgj + Nsj(j) - 1, I6) ^ 2)
Call Cpxalign(ther(i2, I6), thei(i2, I6), sign1, sign2, xx1, xx2)
Call Cpxalign(sigr(i2, I6), sigi(i2, I6), sign3, sign4, xx3, xx4)
List1.AddItem " " &Format(i2, "00") & "=" &sign1 &Format(ther(i2, I6), "0.000000") & "E" & xx1 &sign2 & "j"
&Format(Abs(thei(i2, I6)), "0.000000") & "E" & xx2 & "      " &Format(i2, "00") & "=" & sign3 &Format(sigr(i2,
I6), "0.000000") & "E" & xx3 &sign4 & "j" &Format(Abs(sigi(i2, I6)), "0.000000") & "E" & xx4
Next I6
I4 = 0
Next j
List1.AddItem ""
Next I1
End Sub
```

3.13.28 Sub viewdata_Click( )

```
Private Sub viewdata_Click()
sSTab1.Tab = 0
End Sub
```

3.13.29 Sub viewgraphic_Click( )

```
Private Sub viewgraphic_Click()
sSTab1.Tab = 2
End Sub
```

3.13.30 Sub viewresult_Click( )

```
Private Sub viewresult_Click()
sSTab1.Tab = 1
End Sub
```

3.13.31 Sub savepjt( )

```
Private Sub savepjt(comdlg AsCommonDialog)
Dim Fname As String, I AsInteger, k, vib As String,smsg As String
Label7.Visible = True: Text7.Visible = True
Label8.Visible = True: Text8.Visible = True
Text5.SetFocus
IF Val(Text5.Text) < 0 Then
smsg = MsgBox("Inertiaof Disc(" & Str(s) & ")Invalid!", 48, "MBTV 1.0")
Text5.SetFocus
Exit Sub
End IF
sInn(s, t) = Text5.Text
IF TextRun = 0 Then
sSTab1.Tab = 0
End IF
sSTab1.Tab = 0
```

```
IF Text10.Text = "" Then
smsg = MsgBox("ThereIsn't a project name.Inputone?”& " Then youCan save the project.", 4 + 32 + 256,
"MBTV 1.0")
IFsmsg = 6 Then
Text10.SetFocus
Exit Sub
End IF
End IF
sProname = Text10.Text
sProdate = Str(Time) + "       " + Str(Date)
OnError GoToErrto
Comdlg.CancelError = True
IF TextSave = 0 Then
Comdlg.DialogTitle = "Save"
Comdlg.ShowSave
Fname =Comdlg.FileName
do WhileFname = ""
smsg = MsgBox("Withou aFile name, quit savedata?", 4 + 32 + 256, "MBTV 1.0")
IFsmsg = 6 Then
Exit Sub
Else
comdlg.ShowSave
Fname =Comdlg.FileName
End IF
Loop
End IF
Fname =Comdlg.FileName
openFnameFor output As #2
print #2, ""
print #2, "snalysis Type:Multi-Junction Torsional Vibration with Damping"
print #2, ""
print #2, "project Name:       " & sProname & "by MBTV 1.0 1.0" & "date:    " & sProdate
print #2, "dataFile Name:    " &Fname
print #2, "------------------------------------------------------"
print #2, sProname
print #2, sProdate
IF (js - 1) <= 0 Then
RunFlag = 0
Else
RunFlag = 1
End IF
print #2, Str(RunFlag)
print #2, "Outputdata:"
print #2, sn: Print #2, snj
For I = 1 To nj
print #2,sinjj(i): Print #2, snsj(i)
Next I
For I = 1 To n
print #2,sm(i)
IF Val(sm(i)) = 1 Then
print #2,sinn(i, 1): Print #2, sedp(i, 1)
print #2, sidp(i, 1): Print #2, sk(i, 1)
Else
For k = 1 To Val(sm(i))
print #2,sinn(i, k): Print #2, sedp(i, k)
print #2, sidp(i, k): Print #2, sk(i, k)
Next k
End IF
Next I
IFRunFlag = 0 Then
close:Exit Sub
End IF
print #2, "End data"
IF (js - 1) <= 0 Then
```

```
TextRun = 0:RunFlag = 0:Close:Exit Sub
Else
RunFlag = 1
End IF
Dim I1, I2, I3, I4 AsInteger
I4 = 0
For I3 = 1 To Val(sn)
I4 =I4 + Val(sm(i3))
Next I3
print #2, "Analysis Results"
print #2, Str((js - 1) / 2)
For I1 = 1 To (js - 1) / 2
print #2, Str(omg3(i1))
For I2 = 1 ToI4
print #2, Str(stt1(i1, I2))
Next I2
Next I1
TextSave = 1: TextChang = 0
Close
Errto:Exit Sub
End Sub
3.13.32 Sub saveresults1( )
Private Sub saveresults1()
Dim I AsInteger, vib As String,smsg As String
Dim I1 AsInteger, I2 AsInteger, I3 AsInteger, I4 AsInteger, I5 AsInteger
Dim I6 AsInteger, I7 AsInteger, I8 As Double, I9 AsInteger, I10 AsInteger
Dim thetaer(1 To ROW) As Double, thetaei(1 To ROW) As Double
Dim sigmaer(1 To ROW) As Double, sigmaei(1 To ROW) As Double
Dim ther(1 To ROW, 1 To ROW) As Double, thei(1 To ROW, 1 To ROW) As Double
Dim sigr(1 To ROW, 1 To ROW) As Double, sigi(1 To ROW, 1 To ROW) As Double
Dim stt(1 To ROW) As Double, sgj AsInteger, j AsInteger
Dim omgg(1 To RO) As Double, jg AsInteger
Dim sign1 As String, sign2 As String, sign3 As String, sign4 As String
Dim xx1 As String, xx2 As String, xx3 As String, xx4 As String
Dim thesr As Double, thesi As Double, sigsr As Double, sigsi As Double
Dim thetaEnd r As Double, thetaEnd i As Double
Dim sigmaEnd r As Double, sigmaEnd i As Double
IF TextRun = 0 Then
smsg = MsgBox("There aredataonly, so previewdata.", 0 + 48, "MBTV 1.0")
End IF
vib = "Analysis Resultsof Multi-Branch Multi-Junction Torsional Vibration with Damping"
Mjdata2.CommonDialog2.DefaultExt = "RTF"
Ffname = App.Path & "\mjfdxx.rtf"
Call RootFD(1, 0, 0, 0, omgg(), jg)
List1.Clear
picture1.Picture = LoadPicture()
openFfnameFor output As #2
print #2, ""
print #2, vib
print #2, "File Name:       ";Ffname
print #2, ""
print #2, "--------------------------------------------------------"
print #2, ""
IF TextRun = 0 ThenClose:Exit Sub
I4 = 0: sgj = 1: thesr = 1: thesi = 1:sigsr = 0:sigsi = 0
For I1 = 1 Tojg
print #2, "          ω" & Str(i1) & "=" &Format(omgg(i1), "####.#######")
List1.AddItem "                          θ                                  ΣT"
For j = 1 To nj
Call HolzerFD(omgg(i1), 1, 0, 0, 0, thetaer(sgj), thetaei(sgj), sigmaer(sgj), sigmaei(sgj), ther(), thei(), sigr(),
sigi(), sgj)
For I2 = sgj + 1 To sgj + nsj(j) - 2
Call HolzerFD(omgg(i1), 1, 0, 0, 0, thetaer(i2), thetaei(i2), sigmaer(i2), sigmaei(i2), ther(), thei(), sigr(), sigi(),
I2)
```

```
I4 =I4 + 1
Call Cpxalign(ther(i2, I3), thei(i2, I3), sign1, sign2, xx1, xx2)
Call Cpxalign(sigr(i2, I3), sigi(i2, I3), sign3, sign4, xx3, xx4)
print #2, " " &Format(i2, "00") & "=" &sign1 &Format(ther(i2, I3), "0.000000") & "E" & xx1 &sign2 & "j"
&Format(Abs(thei(i2, I3)), "0.000000") & "E" & xx2 & "      " &Format(i2, "00") & "=" & sign3 &Format(sigr(i2,
I3), "0.000000") & "E" & xx3 &sign4 & "j" & Format(Abs(sigi(i2, I3)), "0.000000") & "E" & xx4
Next I3
End IF
Next I2
Call ReviseFD(omgg(i1), 1, 0, 0, 0, thetaer(), thetaei(), sigmaer(), sigmaei())
Thesr =thetaer(sgj + nsj(j) - 2): thesi =thetaei(sgj + nsj(j) - 2)
For I5 = 1 To sgj + nsj(j) - 2
sigsr =sigsr +sigmaer(i5):sigsi =sigsi +sigmaei(i5)
Next I5
For I6 = 1 To m(sgj + nsj(j) - 1)
I4 =I4 + 1
Call HolzerFD(omgg(i1), thesr, thesi, sigsr, sigsi, thetaEnd r, thetaEnd i, sigmaEnd r, sigmaEnd i, ther(), thei(),
sigr(), sigi(), sgj + nsj(j) - 1)
Thesr =thetaEnd r:sigsr =sigmaEnd r: thesi =thetaEnd i
sigsi =sigmaEnd i: sgj = sgj + nsj(j)
Call Cpxalign(ther(i2, I6), thei(i2, I6), sign1, sign2, xx1, xx2)
Call Cpxalign(sigr(i2, I6), sigi(i2, I6), sign3, sign4, xx3, xx4)
print #2, " " &Format(i2, "00") & "=" &sign1 &Format(ther(i2, I6), "0.000000") & "E" & xx1 &sign2 & "j"
&Format(Abs(thei(i2, I6)), "0.000000") & "E" & xx2 & "      " &Format(i2, "00") & "=" & sign3 &Format(sigr(i2,
I6), "0.000000") & "E" & xx3 &sign4 & "j" & Format(Abs(sigi(i2, I6)), "0.000000") & "E" & xx4
Next I6
I4 = 0
Next j
print #2, ""
Next I1
End Sub
```

3.16 select

This program build selecting vibration analysis type Interface and realize Its Function

3.16.1 Sub Command2_Click( )

```
Private Sub Command2_Click()
Dim smsg As String
IF TextOption1 = 0or TextOption2 = 0 Then
smsg = MsgBox("Need to select Torsional Vibration System and Analysis Type!", 0 + 0 + 48, "MBTV 1.0")
Exit Sub
End IF
RunFlag = 0
Indx = TextOption1 * TextOption2
selectCaseIndx
Case 5
MBTV.select.Hide:MBTV.ildata1.Show:ildata1.SSTab1.Tab = 0
Case 6
MBTV.select.Hide:MBTV.ildata2.Show:ildata2.SSTab1.Tab = 0
Case 7
MBTV.select.Hide:MBTV.ildata3.Show:ildata3.SSTab1.Tab = 0
Case 8
MBTV.select.Hide:MBTV.ildata4.Show:ildata4.SSTab1.Tab = 0
Case 10
MBTV.select.Hide:MBTV.mbdata1.Show:mbdata1.SSTab1.Tab = 0
Case 12
MBTV.select.Hide:MBTV.mbdata2.Show:mbdata2.SSTab1.Tab = 0
Case 14
MBTV.select.Hide:MBTV.mbdata3.Show:mbdata3.SSTab1.Tab = 0
Case 16
MBTV.select.Hide:MBTV.mbdata4.Show:mbdata4.SSTab1.Tab = 0
Case 15
MBTV.select.Hide:MBTV.mjdata1.Show:mjdata1.SSTab1.Tab = 0
Case 18
MBTV.select.Hide:MBTV.mjdata2.Show:mjdata2.SSTab1.Tab = 0
Case 21
```

```
MBTV.select.Hide:MBTV.mjdata3.Show:mjdata3.SSTab1.Tab = 0
Case 24
MBTV.select.Hide:MBTV.mjdata4.Show:mjdata4.SSTab1.Tab = 0
End Select
End Sub
```

3.16.2 Sub Command3_Click( )

```
Private Sub Command3_Click()
Dim I AsInteger,smsg As String
smsg = MsgBox("Close the programe ?", 1 + 32 + 256, "MBTV 1.0")
IFsmsg = 1 ThenEnd
End Sub
```

3.16.3 Sub Form_Load( )

```
Private Sub Form_Load()
MBTV.select.Left = (Screen.Width -MBTV.select.Width) / 2
MBTV.select.Top = (Screen.Height -MBTV.select.Height) / 2
TextSave = 0:RunFlag = 0
End Sub
```

3.16.4 Sub Form_Paint()

```
Private Sub Form_Paint()
Call MBoption3
End Sub
```

3.16.5 Sub option1_Click(index AsInteger)

```
Private Sub option1_Click(index AsInteger)
selectCaseIndex
Case 0
option1(0).Value = True:TextOption1 = 1:Call MBoption1
Case 1
option1(1).Value = True:TextOption1 = 2:Call MBoption2
Case 2
option1(2).Value = True:TextOption1 = 3:Call MBoption3
End Select
End Sub
```

3.16.6 Sub option2_Click(index AsInteger)

```
Private Sub option2_Click(index AsInteger)
selectCaseIndex
Case 0
option2(0).Value = True:TextOption2 = 5:Call MBoption4
Case 1
option2(1).Value = True:TextOption2 = 6:Call MBoption5
Case 2
option2(2).Value = True:TextOption2 = 7:Call MBoption6
Case 3
option2(3).Value = True:TextOption2 = 8:Call MBoption7
End Select
End Sub
```